Contents

1 Welcome to abmSHARE documentation **1**
1.1 What is abmSHARE? 1
1.2 How to become a user? 1

1 Welcome to abmSHARE documentation

abmSHARE is an agent-based (also called individual-based or microsimulation) epidemiological spatial model adapted to SHARE and Eurostat data.

The abmSHARE model is restricted for internal use in the Horizon SHARE-COVID19 project.

1.1 What is abmSHARE?

The abmSHARE model is built on Covasim (COVID-19 Agent-based Simulator), an open-source agent-based model developed by the Institute for Disease Modeling available at `https://github.com/InstituteforDiseaseMod` `covasim`, and build with related SynthPops, an open-source synthetic population constructor developed also by Institute for Disease Modeling available at `https://github.com/InstituteforDiseaseModeling/synthpops`.

The model is written in python and can be adapted by users to suit their research questions and local context by specifying detailed data on population (age structure, mobility, contacts) and the epidemic (diagnosed cases, hospitalization, deaths). As in the original Covasim model, abmSHARE cab be used to explore theoretical research questions or to make projections, its main purpose is to evaluate the effect of different interventions on the epidemic. These interventions include physical interventions (mobility restrictions and masks), diagnostic interventions (testing, contact tracing, and quarantine), and pharmaceutical interventions (vaccination).

abmSHARE extends the original Covasim model in two important dimensions: it allows epidemiological simulations across geographic regions (countries, NUTS) and is able to input external data as parameters for characterizing these regions in terms of their population, policies, or socio-economic conditions, for example. The abmSHARE includes all information of the Covasim model in region-specific geospatial environment on age structure and population size in each region; realistic transmission networks in different social layers, including households, schools, or workplaces; age-specific epidemiological outcomes; and viral dynamics and transmissibility. The abmSHARE model allows for region-specific policy interventions such as non-pharmaceutical interventions (physical distancing, lockdowns), vaccinations, testing, contact tracing and quarantine, all with detailed time structure and other factors.

1.2 How to become a user?

The abmSHARE model has been built for the Horizon SHARE-Covid19 project. To become a user, please contact the administrators of the project for access and support at `radim.bohacek@gmail.com`.

Contents

1 Installation and Remote Access **1**
- 1.1 Installation at your desktop computer . . . 1
- 1.2 Connection to server and sandbox folder structure . . . 1
- 1.3 The sandbox folder . . . 1
  - 1.3.1 Structure of the sandbox folder . . . 1
  - 1.3.2 Input data folder . . . 2
  - 1.3.3 Outputs folder . . . 3
  - 1.3.4 Other individual files . . . 4

1 Installation and Remote Access

1.1 Installation at your desktop computer

1.2 Connection to server and sandbox folder structure

To run abmSHARE remotely, you will need a login and a password. Please contact the administrators for access and support.

1. In a terminal window, login to the server via ssh.
2. Locate sandbox in your home directory: `/home/"username"/sandbox`
3. The sandbox folder contains all necessary files for running abmSHARE. Default configuration can be modified in several configuration and parameter files. Several examples are presented in the Tutorial.
4. For any specification, run the model by typing `./start.sh`
5. All results are stored in the output folder `sandbox/outputs`

1.3 The sandbox folder

The sandbox folder contains all the necessary files for running abmSHARE. The folder is updated for all users after each system update. Previous version of the system are stored with the system update date appended as `sandbox_[date]`. The sandbox folder always contains the updated settings. Please check new/old keys in new and old configuration files for possible changes. Changes are documented in the Changelog page.

1.3.1 Structure of the sandbox folder

```
sandbox
    |
    +-- input_data
    |   +-- data
        |   +-- Czechia.json
        |   +-- NUTS2_mobility_data.csv
        |   +-- population_age_distribution.csv
    |   +-- simulation_configuration_files
        |   +-- simulation_interventions.csv
        |   +-- simulation_global_pars.csv
        |   +-- simulation_region_pars.csv
        |   +-- simulation_vacciness.csv
        |   +-- simulation_variants.csv
    |   +-- simulation_immunity_files
        |   +-- immunity_input_files.csv
        |   +-- vaccine_dose_pars.csv
        |   +-- vaccine_variant_pars.csv
        |   +-- variants_cross_immunity.csv
        |   +-- variants_pars.csv
    |   +-- synthpops_configuration_files
        |   +-- pars_file.csv
        |   +-- synthpops_input_files.csv
        |   +-- synthpops_region.csv
```

```
|   +-- synthpops_input_data
|   |   +-- Various files in here
|   +-- main_configuration.json
|   +-- report_configuration.json
|   +-- main_configuration.json
|   +-- simulation_configuration.json
|   +-- synthpops_configuration.json
+-- logs
+-- outputs
|
+-- parameters.py
+-- results.sh
+-- start.sh
```

1.3.2 Input data folder

This folder contains all configuration files for each module as well as all input data needed for the abmSHARE model to run. These inputs are defined in `json` and/or `csv` files.

Configuration files

- `main_configuration.json` file contains all important parameters for all other modules. More information can be found here.
- `synthpops_configuration.json` file is the synthetic population module configuration file responsible for creating synthetic populations that can serve as inputs into the abmSHARE model. More information can be found here.
- `simulation_configuration.json` file defines the main features of the simulation. It can specify the population created by synthetic population model. More information about settings of this module can be found here.
- `report_configuration.json` file creates outputs of each simulation in csv format as well as a summary of the simulation. More information about this module can be found here.
- Other files are available at download page.

Data folder

Folder name: *data* This folder contains three files by default, which can be shared across simulation and synthetic population module.

- `Czechia.json` contains the geographic data for Czechia. It is used for creating synthetic populations and for running simulations.
- `NUTS2_mobility_data.csv` contains the mobility data for Czechia. It is used for creating synthetic populations and for running simulations.
- `population_age_distribution.csv` contains the age distribution data for Czechia. It is used for creating synthetic populations and for running simulations.

Simulation configuration files folder

Folder name: *simulation_configuration_files* These files are used for defining the simulation parameters. They are used in the simulation module. More info about thouse files you can find T5 Simulation.

- `simulation_interventions.csv` contains the interventions for the simulation. It is used for defining interventions in the simulation module.
- `simulation_global_pars.csv` contains the global parameters for the simulation. It is used for defining global parameters in the simulation module.
- `simulation_region_pars.csv` contains the region-specific parameters for the simulation. It is used for defining region-specific parameters in the simulation module.
- `simulation_vaccines.csv` contains the vaccine parameters for the simulation. It is used for defining vaccine parameters in the simulation module.
- `simulation_variants.csv` contains the variant parameters for the simulation. It is used for defining variant parameters in the simulation module.

Simulation immunity files folder

Folder name: *simulation_immunity_files* These files are used for defining the immunity parameters. They are used in the simulation module. More info about thouse files you can find T5.3 Simulation - Immunity and variants.

- `immunity_input_files.csv` contains the input files for the immunity. It is used for defining the input files in the immunity module.
- `vaccine_dose_pars.csv` contains the vaccine dose parameters for the immunity. It is used for defining the vaccine dose parameters in the immunity module.
- `vaccine_variant_pars.csv` contains the vaccine variant parameters for the immunity. It is used for defining the vaccine variant parameters in the immunity module.
- `variants_cross_immunity.csv` contains the cross-immunity parameters for the immunity. It is used for defining the cross-immunity parameters in the immunity module.
- `variants_pars.csv` contains the variant parameters for the immunity. It is used for defining the variant parameters in the immunity module.

Synthetic population configuration files folder

Folder name: *synthpops_configuration_files* These files are used for defining the synthetic population parameters. They are used in the synthetic population module. More info about thouse files you can find Synthetic popilation configuration file and on T4 Creating population.

- `pars_file.csv` contains the parameters for the synthetic population. It is used for defining the parameters in the synthetic population module.
- `synthpops_input_files.csv` contains the input files for the synthetic population. It is used for defining the input files in the synthetic population module.
- `synthpops_region.csv` contains the region-specific parameters for the synthetic population. It is used for defining the region-specific parameters in the synthetic population module.

Synthetic population other specific input data folder

Folder name: *synthpops_input_data* These files are used for defining more specific parameters for synthetic population creator. More info about thouse files you can find on T4.1 Population input data.

1.3.3 Outputs folder

This folder contains several outputs from a simulation. The `outputs` folder can be set as a default folder in Auto save settings, or it can be used for manually saved outputs defined in various configuration files. Each simulation output is saved to a new folder with a time stamp in the folder name `(YY_mm_DD_HH_MM)`

```
|
+-- outputs
    |   +-- Output_2023_01_31_24_00
        |   +-- Configuration
        |   +-- output_reports
        |   +-- pop_configurations
        |   +-- pops
        |   +-- sims
```

Each simulation result folder has its own subdirectories:

- `Configuration` holds all input configuration files used for the simulation.
- `output_reports` contains all generated `csv` and `xlsx` outputs defined in the Report module.
- `pop_configuration` contains configuration files used as inputs in the Synthetic population module for creating population.
- `pops` holds all generated population objects from Synthetic population module. These can be used in Simulation module as *popfile* key.
- `sims` saves instances of simulations for later analysis and exploration. These are `covasim.MultiSim()` objects.

1.3.4 Other individual files

Start file *(file: start.sh)*

After editing all configuration files, a abmSHARE simulation is started by typing `./start.sh` in the command line. It is important to be in the located (`cd`) in the sandbox directory or to insert an absolute path for the `start.sh` file. If you want to validate the input configuration for the model, run `./start.sh -v`. The validation process will check for correct input data types and file paths.

Start file for grid computing *(file: meta_start.sh)*

For large-scale simulations, the abmSHARE can be run on a grid, using `./start_meta.sh` in the command line, again in the appropriate sandbox directory or by using an absolute path. Input configuration and paths can be validated by Input configuration and paths can be validated by `./start_meta -v true`.

There is also an option to specify a different main configuration file by typing `./start_meta.sh -c <path_to_main_config_file>`. In the validation mode, use the flag as `./start_meta.sh -c <path_to_main_config_file> -v true`.

Input and output data for grid computing are stored in `/home/<username>/sandbox/meta`. Outputs files are automatically downloaded every 30 min from remote computing server.

NOTE In case the server asks for a password or gives an error message, type `source ~/.profile`. If the problem persists, please contact the administrator.

Contents

1 Importing and exporting data from the abmSHARE server **1**
- 1.1 What to import . . . 1
- 1.2 What to export . . . 1
- 1.3 How to transfer files . . . 1
  - 1.3.1 SCP . . . 1
- 1.4 Linux Filezilla . . . 2

1 Importing and exporting data from the abmSHARE server

You can import configuration files or export outputs from simulations by downloading/uploading files with an IDE (Integrated Development Environment) or a common file management or file transfer applications.

Each user with access to the server has a defined `<username>` folder. There is no need to change the path in the input or output files. All paths are defined automatically.

1.1 What to import

In your sandbox folders (`default /home/<username>/sandbox`) there is the `input_files` directory which contains the input configuration files for the abmSHARE model and for input data used by synthetic population module.

The main following import path is

- Configuration files path: `/home/<username>/sandbox/input_data`

1.2 What to export

After running a simulation you can export the output to your local machine. The output files are defined in Sandbox output folder.

1.3 How to transfer files

There are several ways how to import and export files depending on your system.

1.3.1 SCP

For importing or exporting data is reccomended to use the SCP protocol.

SCP via WinSCP For Windows OS you can use WinSCP.

SCP via terminal For downloading **single file** from the abmSHARE server to your PC (local host) use

```
scp username@<servername>:/path/to/file/on/server /path/on/localhost/to/copy
```

For downloading **folders** from the abmSHARE server to your PC (local host) use

```
scp -r username@<servername>:/path/to/folder/on/server /path/on/localhost/to/
    copy
```

For uploading **single file** from your PC (local host) to the abmSHARE server use

```
scp /path/on/localhost/to/copy username@<servername>:/path/to/file/on/server
```

For uploading **folders** from your PC (local host) to the abmSHARE server use

```
scp -r /path/on/localhost/to/copy username@<servername>:/path/to/folder/on/
    server
```

1.4 Linux Filezilla

For data transfer in Linux you can use Filezilla application.

Contents

1 How to run an abmSHARE simulation **1**
- 1.1 Geospatial Input Data System . . . 1
- 1.2 A short testing simulation . . . 1
- 1.3 A full scale simulation . . . 1
- 1.4 Recommended configuration . . . 2
- 1.5 Simulating only a specific module . . . 2
  - 1.5.1 Simulating a synthetic population module . . . 2
  - 1.5.2 Simulation module . . . 3
  - 1.5.3 To run the report module . . . 3

1 How to run an abmSHARE simulation

This chapter contains information about different ways of running an abmSHARE simulation.

1.1 Geospatial Input Data System

abmSHARE extends the Covasim model in two important dimensions. First, it can simulate the model across user-specified geographic regions. Second, it can import parameters obtained from external data sources. These parameters determine the behavior of the model in one or more user-specified geographic regions. These regional and parametric specification can be imported in a table format using `csv` or `xlsx` files. Examples of these input files can be seen and downloaded in the Download section.

Note: Do **not** remove non-optional keys from any configuration file or any columns from a csv file from the configuration files. Do **not** rename files. If a file is missing or a column is empty, the model will use default parameters.

1.2 A short testing simulation

Since the full abmSHARE model is computationally demanding, it is better to try the model on a small scale, testing simulation. For the test, you can/must turn on one or more modules (set to `true`) with the appropriate configuration files fully configured. Also you have to specify key `test` to `true` in the Main configuration file. This test run will generate fixes the size of the population to 20,000 (also with mobility).

In the `main_configuration.json` file:

```
"initialize":
    {
        "synthpop_initialize":true,
        "simulation_initialize":true,
        "report_module_initialize":true,
        "test":true
    }
```

1.3 A full scale simulation

For a full scale simulation turn on the first three modules to `true` with the appropriate configuration files fully configured. The `test` atribute is set to `false` (or can be removed from the configuration file):

In the `main_configuration.json` file:

```
"initialize":
    {
        "synthpop_initialize":true,
        "simulation_initialize":true,
        "report_module_initialize":true,
        "test":false
    }
```

1.4 Recommended configuration

For the first time, run a synthetic population first or use a pre-generated population from a previous simulation. To use such an existing population, turn the synthetic population module to `false` and use a pre-generated population for next runs of the Simulation module. In this case, when you already have created population objects (more can be found at T4 Creating population) you need to specify in the Simulation region configuration file values for the key *popfile* which will hold the full path to the pre-generated population file for specific region. Below is an example of the popfile setting in the region configuration file. This configuration is recommended for faster simulation processing (the same population is not generated in each run).

In the `main_configuration.json` file:

```
"initialize":
    {
        "synthpop_initialize":false,
        "simulation_initialize":true,
        "report_module_initialize":true,
        "test":false
    }
```

Key "popfile" setting in the Simulation module configuration

In the following example, in the first two regions, CZ01 and CZ02, the model will use already generated populations. For the other regions, new populations will be crated.

location_code	use	region_parent_name	name	popfile	pop_infected
CZ01	true	Czechia	Region1	/absolute/path/to/popfile0.pop	2
CZ02	true	Czechia	Region2	/absolute/path/to/popfile1.pop	0
CZ03	true	Czechia	Region3		0
CZ04	true	Czechia	Region4		0
CZ05	true	Czechia	Region5		2
CZ06	true	Czechia	Region6		2
CZ07	true	Czechia	Region7		0
CZ08	true	Czechia	Region8		0

1.5 Simulating only a specific module

You are not limited to use all modules for every run. This approach is useful when you already have generated populations as above in Recommended configuration. Also, you can generate new reports with report module after loading files from an already generated simulation. You can also combine those settings for:

- Only Synthetic population module
- Only Simulation module
- Only Report module
- Combination of Synthetic population module + Simulation module
- Combination of Simulation module + Report module
- Full run of all modules

1.5.1 Simulating a synthetic population module

For simulating only a synthetic population module, set the value in Main configuration file for the key `synthpop_initialize` to `true` and other keys to `false`. In this particular case, you need to specify the path to the `main_configuration.json` file:

```
"initialize": {
        "synthpop_initialize": true,
        "simulation_initialize": true,
```

```
        "report_module_initialize": true,
        "test":false
    },
    "synthpops_settings": {
        "filepath": "<Path_to_sandbox>/input_data/synthpops_configuration.json"
    }
```

1.5.2 Simulation module

For running only the Simulation module, in the Main configuration file `simulation_initialize` to `true` and other keys to `false`. Again, the actual path to the configuration file must be provided. As mentioned in Recommended configuration you can run only this setup or in a combination with report module for a faster process after the population was created and defined in Simulation module configuration.

In the `main_configuration.json` file:

```
"initialize":
    {
        "synthpop_initialize":false,
        "simulation_initialize":true,
        "report_module_initialize":false,
        "test":false
    },
"simulation_settings":
    {
        "filepath":"<Path_to_sandbox>/input_data/covasim_configuration.json"
    }
```

1.5.3 To run the report module

You can run the report module for an already completed simulation. Set the value in Main configuration file of the key `report_module_initialize` to `true` and other keys to `false` with the actual path to the configuration file key `report_settings`.

In the `main_configuration.json` file:

```
"initialize":
    {
        "synthpop_initialize":false,
        "simulation_initialize":false,
        "report_module_initialize":true,
        "test":false
    },
"report_settings":
    {
        "filepath":"<Path_to_sandbox>/input_data/report_configuration.json"
    }
```

Contents

1 Creating a Population **1**
1.1 Input data . . . 1
1.1.1 Basic input data for creating synthetic population . . . 1
1.1.2 Additional input data for creating synthetic population . . . 2
1.1.3 Population creation . . . 2
1.1.4 Parallel simulation . . . 2
1.1.5 Full-scale simulation . . . 2

1 Creating a Population

The Synthetic population module is based on IDM SynthPops available at Synthpops, and extends its functionality. There are two submodules responsible for creating population objects usable in the abmSHARE model. Because the population creation process is computationally demanding, there is an option for a parallel run in which each population is created in a separate CPU core.

All possible configurations are described in the Synthetic population configuration file.

1.1 Input data

For creating a synthetic population there are two main input data sections. First one is for creating a population based on the necessary input arguments. The second is for creating a more specific population based on additional input data that can reflect region specific population distributions across multiple parameters.

1.1.1 Basic input data for creating synthetic population

Basic input data consist of

- `pars_file.csv` contains the parameters for the synthetic population. It is used for defining the parameters in the synthetic population module. The file can be downloaded here.
- `synthpops_input_files.csv` contains the input files for the synthetic population. It is used for defining the input files in the synthetic population module. The file can be downloaded here.
- `synthpops_region.csv` contains the region-specific parameters for the synthetic population. It is used for defining the region-specific parameters in the synthetic population module. The file can be downloaded here.

Parameters file (pars_file.csv) Parameters file is a `csv` file that contains the parameters for the synthetic population. It is used for defining the parameters in the synthetic population module

All possible input data are described in the Synthetic population configuration file. Basically it needs the main `.csv` or `.xlsx` configuration file *synthpops_configuration* file. This files can further contain links to four other parameter files that are described in Synthetic population input_data.

More info about it can be found in Synthetic population configuration file.

Example file structure

location	household_method	smooth_ages
Czechia	fixed_ages	1
Default	fixed_ages	1
CZ01	fixed_ages	1

Region pattern file (synthpops_region.csv) Region pattern file is a csv file that contains the region-specific parameters for the synthetic population. It is used for defining the region-specific parameters in the synthetic population module and enabling regions to be created.

All possible input data are described Synthetic population configuration file. Basically it needs the main .csv or .xlsx configuration file *synthpops_region* file. This files can further contain links to four other parameter files that are described in Synthetic population input_data.

More info about it can be found in Synthetic population configuration file.

Example file structure

location_code	use	region_name	...	parent_dirpath	parent_filename	parent_filepath
CZ01	true	Czechia_CZ01	...			path_to/Czechia.json
CZ02	true	Czechia_CZ02	...	path_to_parent_dirpath	Czechia	

1.1.2 Additional input data for creating synthetic population

Additional input data can be used to specify various attributes such as age distribution, employment rate, enrollment rate, school distributions etc.

Those data can be on the global level (specified by `region_parent_name` or default) or on the region level (specified by `location_code`). If there is a region specific data, it will be used instead of the global data. If there is no region specific data, the global data will be used. **Age distribution file is necessary for creating a synthetic population, otherwise you can specify number of agents in pars_file.csv**

For this approach there is a chapter T4.1 Population input data which describes all possible input data for creating synthetic population.

1.1.3 Population creation

Population creation works in two steps. First one is responsible for creating region pattern configuration file and the other one is creating population object based on this region pattern file and based on additional parameters. There are several option to define parameters (in detail its explained in Synthetic pop settings). As its mentioned in section above - if there is any new data for integrating into population creation, they will take effect only when population is re-created with those new input data. Output populations are by default saved as the '*.pop' files, which are used in simulation module.

1.1.4 Parallel simulation

There is a option to run this module in sequence or in parallel, which can utilize more than one CPU core and improve the performance.

1.1.5 Full-scale simulation

Full run needs to have defined a csv/xlsx files in Synthetic pop settings. Those files are available to download here:

- Synthetic population base region info
- Synthetic population parameters file
- Synthetic population aditional input files description file
- Synthetic population adfitional input files in zip

Contents

1 Input data for creating a synthetic population **1**

1.1 Types of input data . . . 1

1.1.1 Age distribution data . . . 1

1.1.2 region mobility data . . . 2

1.1.3 Employment rate distribution by age . . . 2

1.1.4 School enrollment rate distribution by age . . . 2

1.1.5 Household head age brackets . . . 3

1.1.6 Household head age distribution by family size . . . 3

1.1.7 Household size distribution . . . 3

1.1.8 School size brackets . . . 4

1.1.9 School size distribution . . . 4

1.1.10 School size distribution by type . . . 4

1.1.11 School types by age . . . 5

1.1.12 Workplace size counts by number of people . . . 5

1 Input data for creating a synthetic population

You can create a more specific synthetic population by providing more precise data about the population distribution for regions. This section describes all the input data that can be used for creating a synthetic population. You can create a synthetic population with only a few necessary input data and values. However, if you want to create a more specific synthetic population, you can provide more specific inputs described in this section.

You can specify these specific data globally as described in Synthpops configuration file as a `csv`/`xlsx` file containing of paths to the files. Also you can specify them as 'default','all', or for each region or like 'CZ01' for each parent region name, like 'Czechia'.

NOTE: All keys (column names) must be used in the default format. DO NOT RENAME THEM!

1.1 Types of input data

necessary input data:

- Age distribution

Optional input data:

- Region mobility data
- Employment rate
- Enrollment rate
- Household head age brackets
- Household head age distribution by family size
- School size brackets
- School size distribution
- School size distribution by type
- School types by age
- Workplace size counts by the number of people

1.1.1 Age distribution data

The age distribution contains information about each region and its age distribution. You can use 16, 18 or 20 brackets for the age distribution settings. Also this is the place for providing information about the region code, its name and sum of the population. The distribution of all brackets must sum up to 1.0.

- For 16 brackets: 00_04,...,70_74,75_100
- For 18 brackets: 00_04,...,80_84,85_100
- For 20 brackets: 00_04,...,90_94,95_100

An example can be downloaded here.

location_code	name	population	00_04	05_09	...	95_99
CZ01	Hlavní město Praha	1241664	0.057312606308953	0.041531364362662	...	0.0007804043606
CZ02	Středočeský kraj	1279345	0.063014276836975	0.051952366249917	...	0.000469771640957

1.1.2 region mobility data

Region mobility data can be used for adding people from different regions to a selected region. Basically, it simulates the number of people traveling across regions. These traveling agents are picked randomly from agents traveling from each region. The diagonal is empty, because it is not possible to travel from region to itself.

If the distribution information across regions is not known, you can specify a wider area, for example, a country. So you can use `Czechia` instead of `CZ01`, ..., `CZXX`. However, if you want to specify each region, you can do that as well, or you can also specify only some regions and the rest will be filled with the country level data. **NOTE: for a wider area you have to specify the country name not only in input data, but also in Synthpops region configuration creater**

An example can be downloaded here.

location_code	name	CZ01	CZ02	...	CZ08
CZ01	Hlavní město Praha		15847	...	169
CZ02	Středočeský kraj	100045		...	121

1.1.3 Employment rate distribution by age

Employment distribution data stands for defining the probability of being employed at certain age defined from start year to the end year, commonly in the interval from 16 to 100 years of age. It can also hold region codes for another regions, but it will run only selected based on main synthetic population settings. It can use keys for region codes or region names, optionally. **NOTE: It depends on `region_data_name` defined in region settings, in supplied data must least one region name which correspond with region location code or a wider area name must be supplied**.

An example can be downloaded here.

Age	CZ01	CZ02	...	CZ08	Czechia
16	0.332	0.319	...	0.241	0
17	0.332	0.237	...	0.310	0
...	...	...	...	...	0
100	0.064	0.039	..	0.015	0

1.1.4 School enrollment rate distribution by age

School enrollment distribution data stands for defining the probability of being enrolled in school at certain age defined from start year to the end year, again from 16 to 100 years of age. The same instructions apply for region coding as in the employment rate distribution information above.

An example can be downloaded here.

Age	CZ01	CZ02	...	CZ08	Czechia
0	0	0	...	0	0
...	...	...	...	...	...
17	0.974	0.974	...	0.974	0.974

Age	CZ01	CZ02	...	CZ08	Czechia
18	0.707	0.707	...	0.707	0.707
...	...	...	...	...	...
100	0	0	...	0	0

1.1.5 Household head age brackets

Household head age brackets are used for defining the age brackets for household head. It is used for creating household head age distribution. Every bracket is defined from minimum age to the maximum age, commonly between 15 and 100 years by step of 5. **NOTE: all household related input data must be based on this brackets**

An Example can be downloaded here.

min_age	max_age
15	19
20	24
...	...
75	79
80	100

1.1.6 Household head age distribution by family size

Household head age distribution by family size is used for defining the number of household heads at certain household head age brackets. Each row in this table specifies the distribution for a given family size. The family size is the first entry in the row. The remaining entries are, for each household head age bracket, the number or percentage of households with a household head in that age bracket. In the `*.csv` file the certain household head age brackets are defined by number from 0 to N (based on total number of brackets).

An example can be downloaded here.

number	0	1	2	...	13
1	1.0	1.0	1.0	...	1.0
2	163.0	999.	2316.0	...	2230
...	...	...	...	...	...
7	24.0	33.0	63.0	...	144.0
8	0.0	0.0	0.0	...	0.0

1.1.7 Household size distribution

Household size distribution is used for defining the distribution of households at certain household size. Each row in this table specifies the distribution for a given household size. It depends on the household head age brackets. Every row is defined as the percentage distribution across given every row of household head age bracket. **NOTE: this household input data depends on household head age brackets. Also the sum of all distributions must be equal to 1**.

An example can be downloaded here.

number	distribution
1	0.064
2	0.108

number	distribution
...	...
7	0.128
8	0.0

1.1.8 School size brackets

School size brackets are used for defining the size of school. It is used for creating school size distribution. Every bracket is defined from the minimum to the maximum size. **NOTE: all school related input data must be based on these brackets**

An example can be downloaded here.

min	max
20	50
51	100
101	300
...	...
2301	2700

1.1.9 School size distribution

School size distribution is used for defining the number of schools at certain school size brackets. Each row in this table specifies the distribution for a given school size. It depends on the school size brackets. Every row is defined as the percentage distribution across given every row of the school size bracket.

An example can be downloaded here.

distribution
0.06024096385542162
0.07831325301204821
...
0.006024096385542101
0.0

1.1.10 School size distribution by type

School size distribution by type is used for defining the distribution of school size by school type. Each row in this table specifies the distribution for a given school type. It depends on the school size brackets. Every row is defined as the percentage distribution across given every row of school size bracket. You can specify types of school by yourself and set them age distribution in school types by age section. **NOTE: this school input data depends on school size brackets and on school types by age. Also every row distribution sum must be equal to 1**.

`pk` = preschool, `es` = elementary school, `ms` = middle school, `hs` = high school, `uv` = university

An example can be downloaded here.

school_type	distribution
pk	”0.012658227848101266, 0.0, ..., 0.43037974683544306, 0.0, 0.0, 0.0, 0.0, 0.0,”
es	”0.012658227848101266, 0.0, ..., 0.43037974683544306, 0.0, 0.0, 0.0, 0.0, 0.0,”
hs	”0.07407407407407407, 0.1111111111111111, ... , 0.18518518518518517, 0.0, 0.0 , 0.0”

school_type	distribution
uv	”0.027522935779816515, 0.009174311926605505, ... , 0.2018348623853211, 0.3944954128440367, 0.0”

NOTE: distribution must be in double quotes to be validly parsed

1.1.11 School types by age

School types by age is used for defining the age distribution for each school type. It is used for creating school age distribution. Every row in this table specifies the distribution for a given school type. You can define here those school types.

`pk` = preschool, `es` = elementary school, `ms` = middle school, `hs` = high school, `uv` = university

An example can be downloaded here.

school_type	min_age	max_age
pk	3	5
es	6	10
hs	15	18
uv	19	100

1.1.12 Workplace size counts by number of people

Workplace size counts is defined by number of people in a specific workplace and the total count of those workplaces. There are defined ”brackets” for minimum to maximum agents per workplace and the total count of those workplaces.

An example can be downloaded here.

min_people	max_people	count
1	4	107682.0
5	9	35584.0
...	...	...
500	999	245.0
1000	1999	150.0

Contents

1 Creating a Simulation **1**
- 1.1 How it works . . . 1
  - 1.1.1 Core features . . . 1
- 1.2 Parameters . . . 2
  - 1.2.1 Global parameters . . . 2
  - 1.2.2 Region-specific parameters . . . 3
- 1.3 How to run a simulation . . . 3
  - 1.3.1 Complex run with a synthetic population . . . 3
  - 1.3.2 Complex run without synthetic population run . . . 3

1 Creating a Simulation

1.1 How it works

The simulation module is based on the Covasim model developed by the Institute for Disease Modeling available at `https://github.com/InstituteforDiseaseModeling/covasim` and extends its functionality without changing the core model. This module is invoked by enabling the option *key:simulation_initialize* in the Main configuration settings and defined by its own configuration settings, which are in detail described at Simulation module configuration.

Example configuration and input files for download

- Configuration file for simulation module
- Global parameters csv file
- Region parameters csv file
- Intervention parameters csv file
- Variants parameters csv file

All possible configurations for this module are in Simulation module configuration.

1.1.1 Core features

- **Running a simulation**

The main purpose of the abmSHARE model is to run a simulation of the Covasim model across several geographic regions specified in the input data. For example, you can run it in one country (Czechia with 8 NUTS 2 regions) or in Europe with N countries. Each region can be defined separately in terms of its own parameters (see Different parameters) or all or some regions can share some or all parameters (see Global parameters). Also you can combine some of the global parameters and include different parameters for only some regions. Of course, the model allows for a simulation of a single region or one country.

- **Mobility**

The abmSHARE model allows for the possibility of allowing a number of agents travelling across regions in multisim. This enables the user to track how the illness is spreading across regions during the simulation. An example of the mobility dataset can be downloaded from a Download page. When the mobility feature is turned on and you want to use a pregenerated synthetic population object, you have to use **pregenerated synthethic population which includes mobility** described in T4 Creating population and Synthetic population module. You cannot run simulation with mobility feature and use pregenerated synthetic population object without the mobility feature.

The mobility feature works such that a number of agents who travel across regions are added to that region where their mobility is specified so that they have connection in both regions. After every simulation step they are synchronized; therefore, they can spread the infection in home region as well as in other region.

- **Interventions**

Interventions represent a feature for defining an intervention into the model at a specific day of simulation. For now there are two ways for defining intervention. They can be defined globally for all regions, or differently for a subset of regions.

One new intervention was introduced based on new mobility feature:

- You can specify a quarantine for one or more regions by disabling mobility from and to the selected region(s) for a given time period (date from - date to).

Please see more for interventions in T5.1 Simulation Interventions.

- **Disease variants**

Variants can be used with the default settings of the simulation module described in Parameters original simulation variants. Users can redefine and add their own variants according to T5.3 Simulation - Immunity and variants.

In multisimulation, a user can include the original Covasim disease variants described in Parameters original simulation variants and update its parameters on your own. Variants can be defined globally for each region of the multisimulation or differently for each subset of regions.

Variants must be defined in its own file in the simulation configuration file.

```
  "variants": {
    "filepath": "/home/user/sandbox/input_data/simulation_configuration_files/
        simulation_variants.csv"
  },
```

location_code	variant	use	label	days	start_day	end_day	num_days	number_of_imports
global	omicron	True	omicron_variant				[5,90]	100

1.2 Parameters

You can specify parameters for each simulation separately or define them globally for each region in multisimulation, or combine them. Most parameters are based on the original Covasim model, but also contains some custom parameters. For all parameters see Simulation parameters. Below is the description how to define parameters globally for all multisimulation or separately for each simulation.

1.2.1 Global parameters

Parameters can be defined globally. That means you can choose parameters and assign them value to take effect in every region of of the multisimulation. This approach can allow the parameters to be defined conveniently only in one place in Simulation_settings global parameters. Please see the detailed description there. For global parameters turn the *key: different_pars value* to `false`.

Usage For defining parameters you have to write them into the *simulation_global_pars.csv* file which is described here. Global parameters takes effect in all regions of the simulation. So it is necessary to share the same `start_day`, number of days to simulate and also you can use other parameters, which are described in Parameters. Also you can specify various other parameters, which are available for the **Parameters for Simulation** in Parameters, but be carefull with the parameters which are location specific.

Definition of parameters in the configuration file

```
...
"global_parameters":
{
    "filepath":"/<Path_to_sandbox>/input_data/data/simulation/
        simulation_global_pars.csv"
}
...
```

Definition of global parameters in a `.csv` file saved in the file path in *global_parameters*

n_days	start_day	use_waning	unique_mobility_indexes
90	2020-01-1	true	True

1.2.2 Region-specific parameters

You can define region-specific parameters along with global parameters. In this case, parameters can be specified differently for each region *key:pars* section. Also region specific input file has few necessary keys to describe simulation.

How to define input `csv`/`xlsx` file in the simulation configuration

```
{
    "region_parameters":
    {
        "filepath":"/<Path_to_sandbox>/input_data/data/simulation/
            simulation_region_pars.csv"
    },
}
```

location_code	use	region_parent_name	name	popfile	pop_infected
CZ01	true	Czechia	Region1		2
CZ02	true	Czechia	Region2		0
CZ03	true	Czechia	Region3		0
CZ04	true	Czechia	Region4		0
CZ05	true	Czechia	Region5		2
CZ06	true	Czechia	Region6		2
CZ07	true	Czechia	Region7		0
CZ08	true	Czechia	Region8		0

As you can see, this regions has only `pop_infected` parameter as the additional parameter, which will be also combined with global parameters defined in the global parameters file. Note: This approach is the most effective because most often there will be only a few region-specific parameters.

Note: If some parameters are outside their valid range, a warning is issued. Always check the parameters section

1.3 How to run a simulation

As explained in T3 Running a simulation, you can run an abmSHARE simulation with all combinations described in T3. It is also recommended to enable *key:parallel_run* to take advantages of parallel population object loading.

1.3.1 Complex run with a synthetic population

A full run simulation of all modules automatically loads newly created synthetic population object in multisimulation.

1.3.2 Complex run without synthetic population run

To use the already created synthetic population objects, turn off synthetic population module as in Recommended combination. In this case, a path to the population objects for each region must be specified in *key:popfile*.

Contents

1 Interventions **1**

1.1 How to define an intervention . . . 1
1.2 Intervention timeline settings . . . 1
1.3 Settings for beta interventions . . . 2
1.4 Definition of intervention region . . . 2
1.5 Definition of multiple interventions . . . 3

1 Interventions

For defining special interaction in simulation such as contact tracing, testing agents, or region lockdowns, you can use Covasim interventions. Intervention can be stacked, so there can be the same intervention defined several times with different parameters such as different start day of the intervention, changes across layers affected by the intervention, or other different values related to a specific intervention. More information about interventions can be found in T5.2 Simulation - Intervention types.

As other simulation parameters, an intervention can be defined in several ways: globally for all regions, for each region of the same `region_parent_name` (for example, *Czechia*), or separately for each region of the simulation.

Note: Some interventions are dependent on a different intervention as mentioned in their description

Interventions are defined in `*.csv` or `*.xlsx` file, which contains the correct attributes (columns) for each intervention and valid data. You can define multiple interventions and turn them on/off by *key:use* in the file. It can be also defined as global/state or it can be region-specific.

Interventions are defined in Simulation intervention file.

1.1 How to define an intervention

Specific interventions with basic information and requirements are available in T5.2 Simulation - Intervention types section. The core element is to have a proper `csv`/`xlsx` file with correctly defined attributes. Necessary attributes for each intervention are:

- **location_code** *(str)* related to region/country or global (states are based on *key:region_parent_name* which is defined in Region specific parameters)
- **use** *(bool)* `true` for an intervention, `false` if it is to be ignored
- **label** *(str)* for the intervention, user defined
- **intervention_type** *(str)* such as `mobility_change`, `beta_change` etc. described in Intervention types

An example of intervention attributes:

location_code	use	label	intervention_type	start_day	num_days	end_day	beta_change
CZ01	true	mob_change	mobility_change	2020-03-01		2020-05-19	
Czechia	false	mob_change	mobility_change		[10,25]		
global	true	beta_change	mobility_change		[10,25]		”[0.5]”

1.2 Intervention timeline settings

For the most user-friendly experience there is a complex method that calculates the correct timeline of the intervention. You can supply information about starting/ending time of intervention in two ways:

- Format for date time is in ”YYYY-MM-DD” format
- Format for number of days (takes effect from the start date of the simulation) is integer.

You can combine starting and ending definitions like in the example for a simulation defined in `simulation_global_pars` file:

- **start_day** : ”2020-01-01”, **end_day** : ”2020-01-25”
- **start_day** : 10, **end_day** : 25
- **start_day** : ”2020-01-01”, **num_days** : 25|”[25]”
- **num_days** : ”[1,25]”
- **start_day** : 1, **num_days** : ”2020-01-25”
- **num_days** : 1|”[10]”. **end_day** : ”2020-01-25”

Examples above are all possible combination for calculating the exact same timeline of intervention, which will be from 2020-01-01 to 2020-01-25.

1.3 Settings for beta interventions

Beta changes in specific intervention can be set as a list of values, or as a single value. When single value is used, it can be used for whole intervention period, or defined as the starting point for the whole simulation period. When list of values is used, it must be the same length as the number of days of intervention.

Examples:

- **beta_change** : 0.7, start_day : *defined*, end_day : *defined* -> beta will be 0.7 since start day of intervention period, after end day, it returns to 1
- **beta_change** : 0.7, start_day : *defined*, end_day : *not defined* -> beta will be 0.7 since start day of intervention period for the rest of simulation period
- **beta_change** : [0.7,0.9], start_day : *defined*, end_day : *defined* -> beta will be 0.7 since start day of **intervention** period, after that it will be 0.9 for the rest of intervention period, after end day, it returns to 1
- **beta_change** : [0.7,0.9], start_day : *not defined*, end_day : *defined* -> beta will be 0.7 since start day of **simulation**, after end day it will be set to 0.9 for the rest of simulation period

1.4 Definition of intervention region

There are three ways to define intervention in specific regions. Also you can defined multiple intervention, and enable only a selected ones by explicitly setting the *use* parameter to *true/false*:

Example for a region defined in `csv`:

location_code	use	region_parent_name	name	popfile	pop_infected
CZ01	true	Czechia	Region1	/absolute/path/to/popfile0.pop	2

Example for a region defined by `location_code`:

location_code	use
CZ01	true

Example for an intervention to all regions enabled in the simulation:

location_code	use
global	true

Example for regions defined by the *region_parent_name* in the main region `csv` file:

location_code	use
Czechia	true

1.5 Definition of multiple interventions

In the intervention file you can define multiple intervention by simply adding a column/keys/attributes for each of them and indicating only keys which are correspond to the intervention type. For example, if you want to define two interventions, one for mobility and one for a beta change:

location_code	use	label	intervention_type	start_day	num_days	en
CZ01	true	Region lockdown CZ01	mobility_change	2020-03-01		20
CZ02	true	Region lockdown CZ02	mobility_change		[10,25]	
global	true	Beta change for specific time	beta_change	2020-01-01		20
global	true	Beta change from start to end of simulation	beta_change	2020-01-10	15	

Contents

1 Intervention types **1**
- 1.1 Mobility intervention . . . 1
- 1.2 Change in beta (overall transmission) intervention . . . 1
- 1.3 Contact isolation intervention . . . 1
- 1.4 Daily testing intervention . . . 2
- 1.5 Testing probability intervention . . . 2
- 1.6 Contact tracing intervention . . . 2
- 1.7 Simple vaccination . . . 2
  - 1.7.1 Intervention stacking . . . 3
  - 1.7.2 Requirements . . . 3

1 Intervention types

This section describes the intervention types and their parameters.

Interventions are defined in the proper simulation intervention file

1.1 Mobility intervention

Mobility intervention can be used to simulate region lockdowns. When some region is locked down, agents cannot travel in and out of the region. This intervention can take effect only if **mobility feature** is enabled and defined with proper mobility input data (for details see here).

Example of valid keys for parameters

location_code	use	label	intervention_type	start_day	num_days	end_day
CZ01	true	mob_change	mobility_change	2020-03-01		2020-05-19
CZ02	false	mob_change	mobility_change		"[10,25]"	

1.2 Change in beta (overall transmission) intervention

This intervention can be used to specify transmission by a certain amount for given day/days. Can be used for school closures/restrictions, home office, wearing masks etc. These changes affect the overall transmission parameter in the selected layers.

Example of valid keys for parameters

location_code	use	label	intervention_type	start_day	num_days	end_day	beta_change	layers
global	false	mah	beta_change	2020-01-01		2020-03-19	[0.5]	[w,s]
global	true	meh	beta_change	2020-01-10	15		[0.5,0.7]	

1.3 Contact isolation intervention

Isolate contacts from simulation by, for example, removing contacts for school or for every contact layer. When intervention ends contacts will be restored. Can affect putting people to quarantine, if there is testing and tracking, because of contact removal. Can be used for example for school closures, home offices etc. Compared to the change in beta, contact isolation intervention removes contacts while preserving the transmission value.

Example of valid keys for parameters

location_code	use	label	intervention_type	start_day	num_days	end_day	beta_
CZ01	true	contact isolation interv	isolate_contacts	25		83	[0.7,1

location_code	use	label	intervention_type	start_day	num_days	end_day	beta_
Global	true	contact isolation interv global	isolate_contacts	25			0.7

1.4 Daily testing intervention

Intervention for daily testing specifies the number of agents tested per day. Its value can be also dependent on other values such as the probability of testing people who are quarantined etc. More info can be found in parameters.

Example of valid keys for parameters

location_code	use	label	intervention_type	start_day	num_days	end_day
CZ01	false	Per day testing intervention CZ01	per_day_testing	10		2020-01-30
Global	false	Per day testing intervention global	per_day_testing	2020-01-01		2020-01-30

1.5 Testing probability intervention

This intervention assigns to each person a probability of being tested based on the person's symptom state etc. The total number of tests is not specified as in Daily testing intervention, but it influences the probability of being tested. Can be used as a substitute to the daily testing intervention.

Example of valid keys for parameters

location_code	use	label	intervention_type	start_day	num_days	end_day	symp_
global	false	Testing probability CZ01	testing_probability	20		2020-03-30	0.1
CZ01	false	Tetsing probability CZ01	testing_probability	2020-01-01	30		0.1

1.6 Contact tracing intervention

Contact tracing of agents who are already diagnosed positive by a testing intervention. Contacts are contacted and put in the quarantine (depends on simulation *key:quar_factor* parameter) for a certain period of time. If there is a probability of testing intervention, there can be significant increase of testing of contact-traced agents.

Example of valid keys for parameters. Note that there must be testing intervention (such as Daily testing or Probability testing) before/on day D of starting the contact tracing intervention, otherwise this intervention will not work.

location_code	use	label	intervention_type	num_days	capacity	trace_probs	quar_period
Cz01	false	contact tracing	contact_tracing		0.5	6	6
global	false	contact tracing	contact_tracing	[10,30]	0.5	6	6

1.7 Simple vaccination

Can be used to apply a simple vaccine to a subset of the population. This intervention can be added to changing the relative susceptibility and the probability of developing symptoms if still infected. More info about vaccination can be found in T5.3 Simulation - Immunity and variants.

Example of valid keys for parameters

location_code	use	label	intervention_type	prob	rel_sus	rel_symp
global	true	simple_vac_interv	simple_vaccination	0.9	1.0	1.0

1.7.1 Intervention stacking

You can define multiple interventions and it is not necessary to use all the keys. In the intervention `csv`/`xlsx` file you can define multiple intervention with its corresponding keys and those while leaving empty those interventions that are not used. **Do not define user custom keys which are not corresponding to the allowed keys for each intervention.**

Example of intervention stacking:

location_code	use	label	intervention_type	start_day	num_days	end_day	beta_change	layers
CZ01	true	TestLabel	mobility_change	2020-03-01		2020-05-19		
global	true	TestLabel	beta_change	2020-01-10	15		[0.5,0.7]	
global	false	TestLabel	testing_probability	2020-01-01		2020-03-30		
global	false	TestLabel	contact_tracing	10				
CZ01	false	TestLabel	per_day_testing	10		2020-01-30		
global	false	TestLabel	testing_probability	20		2020-03-30		
global	false	TestLabel	contact_tracing		[10,30]			

1.7.2 Requirements

Valid keys with values must be provided for each intervention, along with the necessary keys.

- Neccessary keys
- Remaining of parameters keys

Contents

1 Reporting module **1**
1.1 Which outputs are generated by default? 1
1.2 How to generate optional outputs? 1
1.3 Recommendations 1

1 Reporting module

Reporting module serves for creating user-friendly data reports from simulations. It is based on outputs generated by the base covasim model. More information about module configuration options can be found at Report_settings

1.1 Which outputs are generated by default?

If this module is enabled, it is responsible for creating the basic output in `*.xlsx` or `*.csv` files. Those files are created for all basic covasim parameters for the whole simulation and for every region separately. Outputs also differentiated for each variant created.

1.2 How to generate optional outputs?

You can also generate more specific output files for selected parameters by defining custom reports in the configuration file. More info about configuration file can be found in Report_settings

1.3 Recommendations

You can use Zeppelin with the pre-generated report files in the report module. Alternatively, you can load the whole output simulation file and use Apache Zeppelin. Of course, any other programming language like R or python can be used for your own research. Finally, you can import the covasim module and use its functions for sophisticated data analysis.

Contents

1 Main configuration file settings **1**
- 1.1 Initialize module settings . . . 1
- 1.2 Auto save settings . . . 1
- 1.3 Settings for Synthetic population module . . . 1
- 1.4 Settings for Simulation module . . . 1
- 1.5 Settings for Report module . . . 1

2 Example configuration json **1**

1 Main configuration file settings

1.1 Initialize module settings

- synthpop_initialize **(bool)** - to enable synthetic population module (creating `jsons` or pop creation)
- simulation_initialize **(bool)** - to enable simulation module
- report_module_initialize **(bool)** - to enable report module for generating `csv`/`xlsx` outputs
- test **(bool) OPTIONAL** - to enable the test mode of a simulation. Recommended approach for debugging or for fast testing (loads 20,000 agents per region)

1.2 Auto save settings

Auto save settings stands for a system for creating smart outputs. **IMPORTANT there cannot be an underscore in the filepath. Please avoid naming of your folders with underscore.**

- value **(bool)** - to enable auto saving
- auto_increment **(bool)** - to enable the same name of output folder with date-time suffix
- dirname **(string)** - to enable the output into named directory
- location **(string)** - path to output directory where all outputs will be saved. In this newly created directory the one specified with `dirname` will be used for all files needed in the simulation
- copy_files **(object)** - object which can hold information about various items to copy, even when the original module was not initialized
- copy_loaded_pop **(bool)** - option for copy population objects, which were created in previous runs and were not created in actual process of the abmSHARE simulation

1.3 Settings for Synthetic population module

- filepath **(string)** - path to Synthetic population configuration file

1.4 Settings for Simulation module

- filepath **(string)** - path to Simulation configuration file

1.5 Settings for Report module

- filepath **(string)** - path to Report configuration file

2 Example configuration json

```
{
    "initialize":
    {
        "synthpop_initialize":true,
        "simulation_initialize":true,
        "report_module_initialize":true
    },
    "auto_save_settings":
    {
```

```
        "value":true,
        "auto_increment":true,
        "dirname":"NoTestValues",
        "location":"/<Path_to_sandbox>/share-covasim/Outputs",
        "copy_files":
        {
            "copy_loaded_pop":true
        }
    },
    "synthpops_settings":
    {
        "filepath":"/<Path_to_sandbox>/data/synthpops_configuration.json"
    },
    "simulation_settings":
    {
        "filepath":"/<Path_to_sandbox>/data/covasim_configuration.json"
    },
    "report_settings":
    {
        "filepath":"/<Path_to_sandbox>/data/test_report_configuration.json"
    }
}
```

Contents

1 Simulation module configuration **1**
- 1.1 Settings for running a simulation . . . 1
- 1.2 Region parameters . . . 1
- 1.3 Global parameters . . . 1
- 1.4 Interventions . . . 2

2 Example configuration `json` **2**

1 Simulation module configuration

There is a list of possible parameters for file `simulation_configuration.json` responsible for creating simulation of defined regions. Example of the configuration file can be found below and is also for download here.

1.1 Settings for running a simulation

- parallel_run: **(bool)** - to enable a parallel simulation that will use as many cores as many sims are defined. Used to load population and initialize simulation much faster.
- test: **(bool) OPTIONAL**- to enable running the simulation in test mode. It will override test settings both for population size and population type also with mobility.

1.2 Region parameters

Input `csv` file responsible for selecting regions to be included in the simulation. There can be multiple regions defined, but it really depends on the *true* value of each row.

Recommended keys are:

- location_code
- use
- region_parent_name
- pop_infected

** Other allowed parameters are described here** Example file for download available here.

Example of `csv`/`xlsx` file structure:

location_code	use	region_parent_name	name	popfile	pop_infected
CZ01	true	Czechia	Region1	/absolute/path/to/popfile0.pop	2
CZ02	true	Czechia	Region2	/absolute/path/to/popfile1.pop	0
CZ03	true	Czechia	Region3		0

1.3 Global parameters

Input `csv` file responsible for defining global parameters for simulation. There can be only one row defined that will be used for all regions. Global parameters can also be defined inside configuration file. In case of same key are defined in both the `csv` and the configuration file, the **`csv` file has priority**.

Recommended keys are here Example file for download is available here.

Example `csv`/`xlsx` file structure:

n_days	start_day	use_waning	unique_mobility_indexes
90	2020-01-1	true	True

1.4 Interventions

Input `csv` file responsible for defining policy interventions during the simulation. There can be multiple interventions depending on the *true* value of each row. More info about intervention can be found here and here. Example file for download available here.

2 Example configuration `json`

```
{
    "parallel_run": true,
    "region_parameters":
    {
        "filepath":"<Path_to_sandbox>/input_data/simulation_configuration_files/
            simulation_region_pars.csv"
    },
    "interventions":
    {
        "filepath":"<Path_to_sandbox>/input_data/simulation_configuration_files/
            simulation_interventions.csv"
    },
    "variants":
    {
        "filepath":""
    },
    "mobility":
    {
        "value":true,
        "filepath":"<Path_to_sandbox>/input_data/simulation_configuration_files/
            NUTS2_mobility_data.csv"
    },
    "population_size":
    {
        "filepath":"<Path_to_sandbox>/input_data/data/population_age_distributions.
            csv"
    },
    "global_parameters":
    {
        "filepath":"<Path_to_sandbox>/input_data/simulation_configuration_files/
            simulation_global_pars.csv"
    }
}
```

Contents

1 Settings for Synthetic population module **1**
1.1 Simulation settings 1

2 Population creation settings **1**
2.1 Pop output naming *(dict)* 1
2.2 Parameters *(pars_file.csv)* 1
2.3 Population region creator *(synthpops_region.csv)* 2
2.3.1 Input data global *(dict)* 3
2.3.2 Mobility data *(dict)* 3

1 Settings for Synthetic population module

There is a list of possible parameters for configuration file **synthpops_configuration.json** responsible for creating the synthetic population as the input for the simulation module. An example of configuration file is below. It can also be downloaded here.

1.1 Simulation settings

The simulation run parameters define how the resulting population objects will be named and created.

- test *(bool)* **Options - true, false** - **OPTIONAL PARAMATER** only for debugging purposes. It sets the maximum size of population to 20,000 agents. If mobility is enabled, it adds 200 more people to the population based on the number of regions.
- parallel_run *(bool)* **Options - true, false** - to run more efficient generation of population objects. It will use one core per region and it is significantly faster than a single-core simulation. **Note: it is recommended to have RAM available at approximately 1 agent= 10kB**

2 Population creation settings

These options define input data and parameters for population creation.

2.1 Pop output naming *(dict)*

- value *(bool)* **Options - true, false** - to change the name of population object, set it to `true` and define the prefix and suffix of the name.
- pop_name_prefix *(str)* **Options - any string** - prefix of the population object name. **Note: Optional**
- pop_name_suffix *(str)* **Options - any string** - suffix of the population object name. **Note: Optional**
- pop_output_type *(str)* **Options - obj/json** - type of the population object to be created. **Note: Obj is much faster and recommended way**
- pop_output_dirpath *(str)* **Options - any string** - path to the directory where population object will be saved. **Note: not needed when autosave_settings enabled**

2.2 Parameters *(pars_file.csv)*

Parameters for defining region parameters. It can be defined as default option, specified by `location_code` or `region_parent_name`. Example: 'default', 'Czechia', 'CZ01'. **NOTE: it is not case sensitive**

Example file for parameters can be found here.

Necessary parameters are: **location_code, household_method, smooth_ages**.

Also you can define optional parameters: *max_contacts, ltcf_pars, shool_pars, with_industry_code, with_facilities, use_two_group_reduction, average_ltcf_degree, with_school_types, school_mixing_type, average_class_size, inter_grade_mixing, average_student_teacher_ratio, average_teacher_teacher_degree,*

teacher_age_min, teacher_age_max, with_non_teaching_staff, average_student_all_staff_ratio, average_additional_staff_degree, staff_age_min, staff_age_max, rand_seed, sheet_name, household_method, smooth_ages, household_method, windows_length.

More info about those additional parameters you can find at the official Synthpops documentation

- filepath *(str)* **Options - any string** - path to the parameters file for population creation. **Note: Necessary**

File structure with different location codes:

location	household_method	smooth_ages
Czechia	fixed_ages	1
Default	fixed_ages	1
CZ01	fixed_ages	1

2.3 Population region creator *(synthpops_region.csv)*

This file is responsible for defining main region characteristics used for population creation. It is location code oriented and it uses various input data for each region. An example file can be found here.

Necessary columns are:

- location_code *(str)* **Options - any string** - location code of the region used for population creation. For example 'CZ01' **Note: Necessary**
- use default *(bool)* **Options - true, false** - to use default parameters for population creation. **Note: Necessary, it also provides and option to hold multiple regions, but to create only a selected ones**
- region_name *(str)* **Options - any string** - name of the region used for population creation. For example 'Czechia_CZ01' **Note: Necessary**
- untrimmed_name *(str)* **Options - any string** - name of the region used for population creation. For example 'Hlavní město Praha' or 'Capital city Prague' **Note: Necessary**
- sheet_name *(str)* **Options - any string** - name of the sheet in the input file used for population creation. For example 'Czech Republic' **Note: Necessary and those codes are based on contact matrices, which you can find here and here the part 2.**
- region_parent_name *(str)* **Options - any string** - name of the parent region used for population creation. For example 'Czechia' **Note: Necessary**
- notes *(str)* **Options - any string** - notes for the region used for population creation. For example 'Capital city' **Note: Optional**
- parent_dirpath *(str)* **Options - any string** - path to the parent directory used for population creation. For example 'CZ01' **Note: Optional**
- parent_filename *(str)* **Options - any string** - name of the parent file used for population creation based on parent_dirpath. For example 'CZ01' **Note: Optional**
- parent_filepath *(str)* **Options - any string** - path to the parent file used for population creation. For example 'CZ01' **Note: Optional**

Note: It is recommended to provide a full path to the parent file, instead of using parent_dirpath and parent_filename.

Note: Parent configuration file is necessary - you can use a default file that can be downloaded here and rename it to your needs. It is used to define all input data if specific input data you can find at synthpops input data are not provided. The best approach is to define a filepath to the file but you can also define a `parent_dirpath` which has multiple parent files and then specify the name of the file in `parent_filename`. The model will calculate and select the file which fits your defined requirements.

File structure with different location codes:

location_code	use	region_name	...	parent_dirpath	parent_filename	parent_filepath
CZ01	true	Czechia_CZ01	...			path_to/Czechia.json
CZ02	true	Czechia_CZ02	...	path_to_parent_dirpath	Czechia	

2.3.1 Input data global *(dict)*

This file is responsible for defining additional input data for population creation. It is location code oriented and it uses various input data for each region. Example file can be found here. It can be defined globally or separately for location codes as: 'Default', Czechia' or 'CZ01'.

More information about these data can be found here and downloaded as zip file here.

The only necessary file is age distribution file.

File structure with different location codes:

location_code	population_age_distribution	...	enrollment_rates_by_age	...
CZ01	path_to_file	...	path_to_file	...
Czechia	path_to_file	...	path_to_file	...

2.3.2 Mobility data *(dict)*

This file is responsible for defining mobility data for population creation. It is location code oriented and it uses various input data for each region. Example file can be downloaded here. Another description can be found here. **Note: mobility data are OPTIONAL**

File structure can be found here.

```
{
    "test":true,
    "parallel_run":true,
    "pop_creator_settings":{
        "pop_output_naming":
        {
            "value":true,
            "pop_name_prefix":"",
            "pop_name_suffix":"",
            "pop_output_type":"obj",
            "pop_output_dirpath":"",
            "pop_creator_dirpath":""
        },
        "parameters":
        {
            "filepath":"<Path_to_sandbox>/sandbox/input_data/data/pars_file.csv"
        },
        "pop_creator":
        {
            "pop_creator_file":
            {
                "value":true,
                "filepath":"<Path_to_sandbox>/sandbox/input_data/data/
                    synthpops_region.csv"
            },
            "input_data_global":
            {
                "value":true,
                "filepath":"<Path_to_sandbox>/sandbox/input_data/data/
                    synthpops_input_files.csv",
                "mobility_data":
                    {
```

```
                            "value":true,
                            "filepath":"<Path_to_sandbox>/sandbox/input_data/data/
                                NUTS2_mobility_data.csv"
                        }
                }
            }
        }
}
```

Contents

1 Report configuration and report module system stands for getting data in chosen format. 1

1.1 Configuration file keys 1

2 Example json configuration file 1

1 Report configuration and report module system stands for getting data in chosen format.

1.1 Configuration file keys

- create_report **(bool)** - This key decides about getting the whole report and calling the report system
- output_format **(string) OPTIONS: csv, xlsx** - this key is for chosen the output data format
- output_path **(string)** - path for saving output files **It can be overwritten, when auto_settings save is enabled**
- input_multisim **(string)** - it serves for more specific defining the input simulation and parameters for output. **It can be overwritten by Auto_saving system, which will load the actual simulation object**
  - filepath **(str)** - filepath to specific sim object.
  - allkeys **(bool)** - if an output files with all the keys should be created
  - keys **(list of strings)** - output file will have only those keys
  - whole_simulation **(bool)** - if output csv should have information about the whole simulation **Can be also combined with separated_simulation**
  - separated_simulation **(bool)** - output csvs will be for every region/simulation separately **Can be also combined with whole_simulation**
  - whole_variants **(bool)** - specifies if user want to create report with variant specific data for all provided sims/regions. **Can be also combined with separated_variants**
  - separated_variants **(bool)** - specifies if user want to create report with variant specific data for every sim/region each separately
- reports **(list of objects)** - reports will be standing for more accurate reports of given variables. **Can also be combined with whole_variants**
  - keys **(list of strings)** - output file will have only those keys
  - output_format **(string) Available OPTIONS: csv, xlsx** - this key is for chosen the output data format **Overrides basic output_format for this report**

2 Example json configuration file

```
{
    "create_report":true,
    "output_format":"csv",
    "input_multisim":
    {
        "filepath":"",
        "all_keys":true,
        "keys":
        [

        ],
        "whole_simulation":true,
        "separated_simulation":true,
        "whole_variants":true,
        "separated_variants":true
    },
    "reports":
    [
        {
```

```
            "keys": ["t","new_deaths","n_dead"],
            "output_format":""
        }
    ]
}
```

Contents

1 Parameters for Synthetic population module **1**

2 Parameters for Covasim variants **2**

3 Parameters for simulation **3**
- 3.1 Admissible parameters for global settings . . . 3
- 3.2 Population parameters . . . 3
- 3.3 Simulation parameters . . . 3
- 3.4 Rescaling parameters . . . 3
- 3.5 Network parameters, generally initialized after the population has been constructed . . . 4
- 3.6 Basic disease transmission parameters . . . 4
- 3.7 Parameters used to calculate immunity . . . 4
- 3.8 Variant-specific disease transmission parameters. . . . 4
- 3.9 Duration parameters: time for disease progression . . . 4
- 3.10 Duration parameters: time for disease recovery . . . 5
- 3.11 Severity parameters: probabilities of symptom progression . . . 5
- 3.12 Efficacy of protection measures . . . 5

4 Intervention parameters **6**
- 4.1 Parameters for mobility intervention . . . 6
- 4.2 Parameters for changing beta intervention . . . 6
- 4.3 Parameters for contact isolation . . . 6
- 4.4 Parameters for daily testing . . . 7
- 4.5 Parameters for testing probability . . . 7
- 4.6 Parameters for contact tracing . . . 8
- 4.7 Parameters for simple vaccination . . . 8

1 Parameters for Synthetic population module

These parameters can be used in Synthetic population settings in *key:pars* section. More information can be found at IDM SynthPops.

Example parameters with default values:

- use_two_group_reduction=True
- average_LTCF_degree=20
- ltcf_staff_age_min=20
- ltcf_staff_age_max=60
- with_school_types=False
- school_mixing_type='random'
- average_class_size=20
- inter_grade_mixing=0.1
- average_student_teacher_ratio=20
- average_teacher_teacher_degree=3
- teacher_age_min=25
- teacher_age_max=75
- with_non_teaching_staff=False
- average_student_all_staff_ratio=15
- average_additional_staff_degree=20
- staff_age_min=20
- staff_age_max=75
- rand_seed=None
- country_location=None
- state_location=None
- location=None
- sheet_name=None
- household_method='infer_ages'

- smooth_ages=False
- window_length=7

2 Parameters for Covasim variants

- name: "wild" *#default*
  - rel_beta = 1.0, # Default values
  - rel_symp_prob = 1.0, # Default values
  - rel_severe_prob = 1.0, # Default values
  - rel_crit_prob = 1.0, # Default values
  - rel_death_prob = 1.0, # Default values
- name: "alpha"
  - rel_beta = 1.67, # Midpoint of the range reported in https://science.sciencemag.org/content/372/6538/eabg3055.
  - rel_symp_prob = 1.0, # Inconclusive evidence on the likelihood of symptom development. See https://www.thelancet.com/journals/lanpub/article/PIIS2468-2667(21)00055-4/fulltext.
  - rel_severe_prob = 1.64, # From https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3792894, and consistent with https://www.eurosurveillance.org/content/10.2807/1560-7917.ES.2021.26.16.2100348 and https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/961042/S1095_NERVTAG_update_note_on_B.1.1.7_severity_20210211.pdf
  - rel_crit_prob = 1.0, # Various studies have found increased mortality for B117 (summary here: https://www.thelancet.com/journals/laninf/article/PIIS1473-3099(21)00201-2/fulltext#tbl1), but not necessarily when conditioned on having developed severe disease
  - rel_death_prob = 1.0, # See comment above
- name: "beta"
  - rel_beta = 1.0, # No increase in transmissibility; B1351's fitness advantage comes from the reduction in neutralisation
  - rel_symp_prob = 1.0,
  - rel_severe_prob = 3.6, # From https://www.eurosurveillance.org/content/10.2807/1560-7917.ES.2021.26.16.2100348
  - rel_crit_prob = 1.0,
  - rel_death_prob = 1.0,
- name: "gamma"
  - rel_beta = 2.05, # Estimated to be 1.7–2.4-fold more transmissible than wild-type: https://science.sciencemag.org/content/early/2021/04/13/science.abh2644
  - rel_symp_prob = 1.0,
  - rel_severe_prob = 2.6, # From https://www.eurosurveillance.org/content/10.2807/1560-7917.ES.2021.26.16.2100348
  - rel_crit_prob = 1.0,
  - rel_death_prob = 1.0,
- name: "delta"
  - rel_beta = 2.2, # Estimated to be 1.25-1.6-fold more transmissible than B117: https://www.researchsquare.com/article/rs-637724/v1
  - rel_symp_prob = 1.0,
  - rel_severe_prob = 3.2, # 2x more transmissible than alpha from https://mobile.twitter.com/dgurdasani1/status/1403293582279294983
  - rel_crit_prob = 1.0,
  - rel_death_prob = 1.0,

3 Parameters for simulation

All parameters are written as the key for use and possible values. All assigned values are representing default Covasim parameters. Some of those parameters are not implemented and tested yet - look for their description in each section. TO DO: What does this mean? Some parameters do not need to use, because they are already used by another module.

3.1 Admissible parameters for global settings

- use_waning = True **(bool) - (Optional)** Whether to use dynamically calculated immunity. **Must be enabled if there are vaccine intervention**
- unique_mobility_indexes = True **(bool)** As from the *V0.2.4* this must be enabled. Future fix coming.
- start_day = '2020-03-01' **str - (Optional)** Date from when the simulation is started. *Can be combined as start_day and n_days. Do not need to specify end_day.*
- n_days = 180 **int** - number of days for simulation

3.2 Population parameters

- pop_size = (int) e.g. 20000 or 20e3 - Number of agents, **Do not use when there is population object loaded**
- pop_infected = (int) 20 - Number of initial infections - **Can be used Globally or separately**
- pop_type = 'random' - What type of population data to use -- 'random' (fastest), 'synthpops' (best), 'hybrid' (compromise) **Don not use when there is population object loaded**
- location = None - What location to load data from -- default Seattle **Do not use when there is population object loaded**

3.3 Simulation parameters

- start_day = '2020-03-01' - **(Optional)** Date from when is simulation started. *Can be combined as start_day and n_days. Do not need to specify end_day.*
- end_day = None **(Datetime (YYYY-MM-DD))- (Optional)** End day of the simulation - can be used with `start_day`. *Does not need to be specified when there is start_day and n_days*
- n_days = 60 **(int)** - Number of days to run, if `end_day` is not specified. *Simulation can only have n_days and starts with default date and ends after las day from n_days*
- rand_seed = 1 **(int) - (Optional)** Random seed, if None, do not reset *Simulation is using some random values, use rand_seed for different random seed*
- verbose = cvo.verbose - **(Optional)** Whether or not to display information during the run -- options are 0 (silent), 0.1 (some; default), 1 (default), 2 (everything). *It signifanctly slower simulation process*

3.4 Rescaling parameters

Rescaling parameters do not need to be used. Some intervention have problem with rescaling (for example, contact tracing).

- pop_scale = 1 **(int) -(Optional)** Factor by which to scale the population -- e.g. `pop_scale`=10 with `pop_size`=100e3 means a population of 1 million. **Cannot be used with interventions for contact tracing**
- scaled_pop = None - The total scaled population, i.e. the number of agents times the scale factor **No need for use**
- rescale = True **(bool)** - Enable dynamic rescaling of the population -- starts with `pop_scale`=1 and scales up dynamically as the epidemic grows. **Use the default, so no need to explicitly specify**
- rescale_threshold = 0.05 **(float)** - Fraction susceptible population that will trigger rescaling (if rescaling enabled)
- rescale_factor = 1.2 **(float**- Factor by which the population is rescaled on each step
- frac_susceptible = 1.0 **(float)**- What proportion of the population is susceptible to infection

3.5 Network parameters, generally initialized after the population has been constructed

Because of synthetic pre-generated population there is no need to use Network parameters. For more info look for covasim original documentation.

- contacts = None - The number of contacts per layer; set by reset_layer_pars() below
- dynam_layer = None - Which layers are dynamic; set by reset_layer_pars() below
- beta_layer = None - Transmissibility per layer; set by reset_layer_pars() below

3.6 Basic disease transmission parameters

- beta_dist = dict(dist='neg_binomial', par1=1.0, par2=0.45, step=0.01)**(dict)** - **(Optional)** Distribution to draw individual level transmissibility; dispersion from https://www.researchsquare.com/article/rs-29548/v1
- viral_dist = dict(frac_time=0.3, load_ratio=2, high_cap=4) **(dict)** - **(Optional)** The time varying viral load (transmissibility); estimated from Lescure 2020, Lancet, https://doi.org/10.1016/S1473-3099(20)30200-0
- beta = 0.016 **(float)** - **(Optional)** Beta per symptomatic contact; absolute value, calibrated
- asymp_factor = 1.0 **(float)** - **(Optional)** Multiply beta by this factor for asymptomatic cases; no statistically significant difference in transmissibility: https://www.sciencedirect.com/science/article/pii/S1201971220302502

3.7 Parameters used to calculate immunity

Parameters defined as dict are not tested in current version. Its recommended to use only *use_waning, trans_redux, nab_boost* parameter from this section.

- use_waning = True **(bool)** - **(Optional)** Whether to use dynamically calculated immunity. **Must be enabled if there are vaccine intervention**
- nab_init = dict(dist='normal', par1=0, par2=2) ** - **(Optional)** Parameters for the distribution of the initial level of log2(nab) following natural infection, taken from fig1b of https://doi.org/10.1101/2021.03.09.21252641
- nab_decay = dict(form='nab_growth_decay', growth_time=21, decay_rate1=np.log(2) / 50, decay_time1=150, decay_rate2=np.log(2) / 250, decay_time2=365)**(dict)** **(Optional)**
- nab_kin = None - **(Optional)** Constructed during sim initialization using the nab_decay parameters
- nab_boost = 1.5 - **(Optional)** Multiplicative factor applied to a person's nab levels if they get reinfected. No data on this, assumption.
- nab_eff = dict(alpha_inf=1.08, alpha_inf_diff=1.812, beta_inf=0.967, alpha_symp_inf=-0.739, beta_symp_inf=0.038, alpha_sev_symp=-0.014, beta_sev_symp=0.079)**(dict)** - **(Optional)** Parameters to map nabs to efficacy
- rel_imm_symp = dict(asymp=0.85, mild=1, severe=1.5)**(dict)** -**(Optional)** Relative immunity from natural infection varies by symptoms. Assumption.
- immunity = None - **(Optional)** Matrix of immunity and cross-immunity factors, set by init_immunity() in immunity.py
- trans_redux = 0.59 - **(Optional)** Reduction in transmission for breakthrough infections, https://www.medrxiv.org/content/10.1101/2021.07.13.21260393v

3.8 Variant-specific disease transmission parameters.

When there are multiple variants from T5.2 Simulation - Variants then there is no need to use those parameters.

- rel_beta = 1.0 ## Relative transmissibility varies by variant

3.9 Duration parameters: time for disease progression

Duration parameters are not implemented yet

- dur['exp2inf = dict(dist='lognormal_int', par1=4.5, par2=1.5) # Duration from exposed to infectious; see Lauer et al., https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7081172/, appendix table S2, subtracting - inf2sym duration
- dur['inf2sym = dict(dist='lognormal_int', par1=1.1, par2=0.9) # Duration from infectious to symptomatic; see Linton et al., https://doi.org/10.3390/jcm9020538, from Table 2, 5.6 day incubation period - 4.5 day exp2inf from Lauer et al.
- dur['sym2sev = dict(dist='lognormal_int', par1=6.6, par2=4.9) # Duration from symptomatic to severe symptoms; see Linton et al., https://doi.org/10.3390/jcm9020538, from Table 2, 6.6 day onset to hospital admission (deceased); see also Wang et al., https://jamanetwork.com/journals/jama/fullarticle/2761044, 7 days (Table 1)
- dur['sev2crit = dict(dist='lognormal_int', par1=1.5, par2=2.0) # Duration from severe symptoms to requiring ICU; average of 1.9 and 1.0; see Chen et al., https://www.sciencedirect.com/science/article/pii/S0163445320301195, 8.5 days total - 6.6 days sym2sev = 1.9 days; see also Wang et al., https://jamanetwork.com/journals/jama/fullarticle/2761044, Table 3, 1 day, IQR 0-3 days; std=2.0 is an estimate

3.10 Duration parameters: time for disease recovery

Duration parameters are not implemented yet

- dur['asym2rec = dict(dist='lognormal_int', par1=8.0, par2=2.0) # Duration for asymptomatic people to recover; see Wölfel et al., https://www.nature.com/articles/s41586-020-2196-x
- dur['mild2rec = dict(dist='lognormal_int', par1=8.0, par2=2.0) # Duration for people with mild symptoms to recover; see Wölfel et al., https://www.nature.com/articles/s41586-020-2196-x
- dur['sev2rec = dict(dist='lognormal_int', par1=18.1, par2=6.3) # Duration for people with severe symptoms to recover, 24.7 days total; see Verity et al., https://www.thelancet.com/journals/laninf/article/PIIS1473-3099(20)30243-7/fulltext; 18.1 days = 24.7 onset-to-recovery - 6.6 sym2sev; 6.3 = 0.35 coefficient of variation * 18.1; see also https://doi.org/10.1017/S0950268820001259 (22 days) and https://doi.org/10.3390/ijerph17207560 (3-10 days)
- dur['crit2rec = dict(dist='lognormal_int', par1=18.1, par2=6.3) # Duration for people with critical symptoms to recover; as above (Verity et al.)
- dur['crit2die = dict(dist='lognormal_int', par1=10.7, par2=4.8) # Duration from critical symptoms to death, 18.8 days total; see Verity et al., https://www.thelancet.com/journals/laninf/article/PIIS1473-3099(20)30243-7/fulltext; 10.7 = 18.8 onset-to-death - 6.6 sym2sev - 1.5 sev2crit; 4.8 = 0.45 coefficient of variation * 10.7

3.11 Severity parameters: probabilities of symptom progression

- rel_symp_prob = 1.0 **(float)** - **(Optional)** Scale factor for proportion of symptomatic cases
- rel_severe_prob = 1.0 **(float)** - **(Optional)** Scale factor for proportion of symptomatic cases that become severe
- rel_crit_prob = 1.0 **(float)** - **(Optional)** Scale factor for proportion of severe cases that become critical
- rel_death_prob = 1.0 **(float)** - **(Optional)** Scale factor for proportion of critical cases that result in death
- prog_by_age = prog_by_age *or* False **(str/bool)** - Whether to set or not disease progression based on the person's age.
- prognoses = None **(float)** - **(Optional)** The actual arrays of prognoses by age; this is populated later

3.12 Efficacy of protection measures

- iso_factor = None **(float)** - **(Optional)** Multiply beta by this factor for diagnosed cases to represent isolation
- quar_factor = None **(float)** - **(Optional)** Quarantine multiplier on transmissibility and susceptibility
- quar_period = 14 **(int)** - **(Optional)** Number of days to quarantine for; assumption based on standard policies

4 Intervention parameters

4.1 Parameters for mobility intervention

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)** = "mobility_change" # Default value for type is **mobility_change**. iNo other value allowed.
- label **(str)** = "Some label" # **Optional** Label for this mobility change eg. "Region1 lockdown"
- start_day **(str in YYYY-MM-DD)/ int as number of day** = "2021-03-16"/ 16 Specifies day of the intervention start
- end_day **(str in YYYY-MM-DD)int as number of day** = "2021-03-16"/ 16 Specifies day of the intervention end
- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10

4.2 Parameters for changing beta intervention

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)** = "beta_change" # Default value for type is **beta_change**. No other value allowed.
- label **(str)** = "Some label" # **Optional**Label for this mobility change eg. "School lockdown"
- start_day **(str in YYYY-MM-DD)** = "2021-03-16" # Specifies day of the intervention start
- end_day **(str in YYYY-MM-DD)** = "2021-03-16" # Specifies day of the intervention end
- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10 `"days":[1,30]` - which will set starting day as 1st day of simulation and it will end the 30th day of simulation. Or can be used as start day only `"days":10` - intervention will start the 10th day of simulation.
- beta_change **(float/float list)** = change overall beta (transmission) parameter. Can be used as single value `"beta_change":0.6` and after intervention ends it will go back to 1.0. Or can be used as list `"beta_change":[0.6,0.8]` and after intervention ends the value will be se to 0.8
- layers **list of str** = change can be applied to one or more layers. You can use it only for school `"layers":['s']` or for more layers e.g. workplaces `"layers":['s','w']`

4.3 Parameters for contact isolation

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)** = "isolate_contacts" # Default value for type is **isolate_contacts**. No other value allowed.
- label **(str)** = "Some label" **Optional** # Label for this mobility change eg. "School contact isolation"
- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10`"days":"[1,30]"` - which will set starting day as 1st day of simulation and it will end the 30th day of simulation. Or can be used as start day only `"days":10` - intervention will start the 10th day of simulation.
- changes **(float/float list)** = change overall contacts by given number (0.3 == - 30% of contacts). Can be used as single value `"changes":0.6` and after intervention ends it will go back to 1.0. Or can be used as list `"changes":"[0.6,0.8]"` and after intervention ends the value will be se to 0.8
- layers **list of str** = change can be applied to one or more layers. You can use it only for school `"layers":"['s']"` or for more layers e.g. workplaces `"layers":"['s','w']"`
- start_day **(str in YYYY-MM-DD)** = "2021-03-16" # Specifies day of the intervention start
- end_day **(str in YYYY-MM-DD)** = "2021-03-16" # Specifies day of the intervention end

4.4 Parameters for daily testing

There is a plenty of key:values which are optional. Basically its necessary to define daily_test and type of intervention. Everything else is optional and be loaded as default parameters, if none provided. (Default parameters are written on every key = default_value). **Note: do not use dict value specific keys, this is not tested yet e.g. *key:subtarget***

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)** = "per_day_testing" # Default value for type is **per_day_testing**. No other value allowed.
- label **(str)** = "Some label" **Optional** # Label for this mobility change eg. "School contact isolation"
- daily_tests **(int/int list)** = number of tests per day, can be int, array, or dataframe/series; if integer, use - that number every day
- symp_test **(float)** = 100.0 **Optional**odds ratio of a symptomatic person testing (default: 100x more likely)
- quar_test **(float)**= 1.0 **Optional**probability of a person in quarantine testing (default: no more likely)
- quar_policy **(str)** = None **Optional**policy for testing in quarantine: options are 'start' (default), 'end', 'both' (start and end), 'daily'; can also be a number or a function, see get_quar_inds()
- subtarget **(dict)** = None **Optional**subtarget intervention to people with particular indices (format: {'ind': array of indices, or function to return indices from the sim, 'vals': value(s) to apply} **NOT IMPLEMENTED YET**
- ili_prev **(arr)** = None **Optional**prevalence of influenza-like-illness symptoms in the population; can be float, array, or dataframe/series
- sensitivity **(float)**= 1.0 **Optional**test sensitivity (default 100%, i.e. no false negatives)
- loss_prob **(float)**= 0 **Optional**probability of the person being lost-to-follow-up (default 0%, i.e. no one lost to follow-up)
- test_delay **(int)** = 0 **Optional**days for test result to be known (default 0, i.e. results available instantly)
- start_day **(int)** = 0 **Optional**day the intervention starts (default: 0, i.e. first day of the simulation)
- end_day **(int)** = None **Optional**day the intervention ends
- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10

4.5 Parameters for testing probability

There is a plenty of `key:values` which are optional. Basically its necessary to define `daily_test` and type of intervention. Everything else is optional and be loaded as default parameters, if none provided. (Default parameters are written on every key = default_value). **Note: do not use dict value specific keys, this is not tested yet e.g. *key:subtarget* **

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)**= "testing_probability" # Default value for type is **testing_probability**. No other value allowed.
- label **(str)** = "Some label" **Optional** # Label for this mobility change eg. "School contact isolation"
- symp_prob **(float)** = 0.1 probability of testing a symptomatic (unquarantined) person
- asymp_prob **(float)** = 0.0 **Optional** probability of testing an asymptomatic (unquarantined) person (default: 0)
- symp_quar_prob **(float)** = None **Optional** probability of testing a symptomatic quarantined person (default: same as `symp_prob`)
- asymp_quar_prob **(float)** = None **Optional** probability of testing an asymptomatic quarantined person (default: same as `asymp_prob`)
- quar_policy **(str)** = None **Optional** policy for testing in quarantine: options are 'start' (default), 'end', 'both' (start and end), 'daily'; can also be a number or a function, see get_quar_inds()

- subtarget **(dict)** = None **Optional** subtarget intervention to people with particular indices (see test_num() for details) **NOT IMPLEMENTED YET**
- ili_prev **(float/float list)** = None **Optional** prevalence of influenza-like-illness symptoms in the population; can be float, array, or dataframe/series
- sensitivity **(float)** = 1.0 **Optional** test sensitivity (default 100%, i.e. no false negatives)
- loss_prob **(float)** = 0.0 **Optional** probability of the person being lost-to-follow-up (default 0%, i.e. no one lost to follow-up)
- test_delay **(int)** = 0 **Optional** days for test result to be known (default 0, i.e. results available instantly)
- start_day **(int)** = 0 **Optional** day the intervention starts (default: 0, i.e. first day of the simulation)
- end_day **(int)** = None **Optional** day the intervention ends (default: no end)
- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10

4.6 Parameters for contact tracing

There is a plenty of `key:values` which are optional. Basically it is necessary to define `daily_test` and type of intervention. Everything else is optional and be loaded as default parameters, if none provided. (Default parameters are written on every key = default_value). **Note: do not use dict value specific keys, this is not tested yet e.g. *keys: trace_probs and trace_time otherwise use them only as single float value* **

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)*** = "contact_tracing" # Default value for type is **contact_tracing**. No other value allowed.
- label **(str)** = "Some label" **Optional** # Label for this mobility change eg. "School contact isolation"
- trace_probs **(float/dict)**=0.4 **Optional** probability of tracing, per layer (default=100%, i.e. everyone is traced) **NOT IMPLEMENTED YET**
- trace_time **(float/dict)**=0 **Optional** days required to trace, per layer (default=0, i.e. no delay) **NOT IMPLEMENTED YET**
- start_day **(int)**=0 **Optional** intervention start day (default=0, i.e. the start of the simulation)
- end_day **(int)**=None **Optional** intervention end day (default=no end)
- presumptive **(bool)**=false **Optional** whether or not to begin isolation and contact tracing on the presumption of a positive diagnosis (default=no)
- capacity **(int)**=None **Optional** optionally specify a maximum number of newly diagnosed people to trace each day
- quar_period **(int)**=None **Optional** number of days to quarantine when notified as a known contact.
- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10

4.7 Parameters for simple vaccination

There are a few keys to define this intervention. Use waning immunity must be enabled.

- location_code **(str)** = location code of region
- use **(bool)** = True # Default value for use is **True**. If you want to disable this intervention, set it to **False**
- intervention_type **(str)** = "simple_vaccination" # Default value for type is **vaccine**. No other value allowed.
- label **(str)** = "Some label" **Optional** # Label for this mobility change eg. "School contact isolation"
- start_day **(int)** = 0 **Optional** intervention start day (default=0, i.e. the start of the simulation)
- end_day **(int)** = None **Optional** intervention end day (default=no end)

- num_days **(int/list of ints)** = can be specified for starting/ending day. Usage as list or integer: [10,25] or 10
- prob **(float)** = 0.0 **Optional** probability of being vaccinated each day (default=0, i.e. no one is vaccinated)
- rel_sus **(float)** = 0.0 **Optional** relative susceptibility of vaccinated people (0 = perfect, 1 = no effect)
- rel_symp **(float)** = 0.0 **Optional** relative symptomaticity of vaccinated people (0 = perfect, 1 = no effect)
- cumulative **(bool)** = False **Optional** whether cumulative doses have cumulative effects (default false);