AMTraC-19 (v7_7d) user guide

Cameron Zachreson,[1] Sheryl L. Chang,[1] Oliver M. Cliff,[1] and Mikhail Prokopenko[1]

[1] *Centre for Complex Systems, The University of Sydney, Sydney, New South Wales, Australia, 2006.*

I. SUMMARY

This document describes how to simulate COVID-19 pandemics in Australia using the AMTraC-19 simulation platform. Part (A) describes the steps involved in using the population generation component of the platform to convert census data from the Australian Bureau of Statistics (ABS) into surrogate populations for use with the epidemic simulator. Part (B) describes how to set up and run the epidemic simulator, including how to implement various types of Social Distancing intervention strategies.

II. PART A: POPULATION GENERATION

A. Input data

The following datasets are required as inputs into population_generation_2016.cpp (note that the datafiles listed here are for 2016, the code will need to be modified slightly if different input file names are used) :

- AGE.dat - the age group distribution for each census district (Statistical Local Area Level 1). The columns correspond to the following:
  1. column 1: SA1 identifier (7-digit code)
  2. column 2: TOTAL, number of individuals residing in the corresponding SA1
  3. column 3: M0-4, number of males between 0 and 4 years of age
  4. column 4: F0-4, number of females between 0 and 4 years of age
  5. column 5: M5-18, number of males between 5 and 18 years of age
  6. column 6: F5-18, number of females between 5 and 18 years of age
  7. column 7: M19-29, number of males between 19 and 29 years of age
  8. column 8: F19-29, number of females between 19 and 29 years of age
  9. column 9: M30-64, number of males between 30 and 64 years of age
  10. column 10: F30-64, number of females between 30 and 64 years of age
  11. column 11: M64+, number of males over the age of 64
  12. column 12: F64+, number of females over the age of 64

- CDinSLA.dat - a list of correspondences between Statistical Area Level 1 (SA1) and Statistical Area Level 2 (SA2). [Note: prior to 2011, areas on the scale of SA1 were referred to as "Census Districts", and those on the scale of SA2 were referred to as "Statistical Local Areas".]

- DZinSLA.dat - a list of correspondences between Destination Zones (DZN) and SA2s (DZN is the partition used by the ABS for partitioning workplaces).

- HOUSEHOLD_COMPOSITION.dat - the distribution of household composition types for each SA1. These are listed in the following order:
  1. column 1: the SA1 identifier (7-digit code)
  2. column 2: S, single adults
  3. column 3: CWOC, couples without children (two adults)
  4. column 4-7: CWC1-4, couples with from one to four children (two adults, 1-4 children)
  5. column 8-12: SPF1-5, single parent family with one adult and from one to five children (one adult, 1-5 children)
  6. column 13-17: G2-6, non-family groups from with 2 to 6 adults.

- TravelToWork.dat - the list of commuter numbers between SA1 and DZN partitions. This is formatted as follows:
    1. column 1: SA1 identifier (7-digit code)
    2. column 2: DZN identifier (11-digit code)
    3. column 3: number of commuters

- DistanceSLA.txt - the matrix of geographic distances between the centroids of each SA2 pair (this was created using the ArcMAP GIS software platform with ABS-provided shapefiles) - the top row lists the SA2 identifier for each column and row of the distance matrix.

- postcode_enrolment_2016.dat - a list of postcodes and number of enrolled students for reach school (primary and secondary school) in Australia for the year 2016. These are documented by the Australian Curriculum Assessment and Reporting Agency (ACARA), not by the ABS.

- Postcode2016_SA2_2016.dat - a correspondence between postcodes and SA2 partitions (these are not exact correspondences - see Zachreson *et al.* 2018 for a description of how they were generated using ArcMAP to analyse partition overlap).

B. Compiling and running generate_population_2016.cpp

This script must be located in the same directory as debug.h, utils.hpp, enums.h, and generate_population.h. The header file 'generate_population.h' (lines 6 and 9) must be edited with appropriate locations for input and output directories. If new input files are used (i.e., for different census years) the 'generate_population_2016.cpp' script must be edited to read the data in from the correct files.
If these small changes are not made, compilation errors will occur. Otherwise, the code can be compiled from the Linux command line using gcc, specifying C++11 libraries and optimisation:

```
g++ generate_population_2016.cpp -std=c++11 -O2
```

The resulting program can then be executed without input arguments, and will produce command line output related to the population generation process.

C. population generation output files

A successful run of the program will produce the following outputs:

- population.dat* : A $5 \times N$ matrix where each row represents an individual Agent generated by the software and the columns correspond to the following:
    1. Age group - an index between zero and 4, corresponding to the age groups listed above, (index increasing with age).
    2. Household - an index corresponding to the household group to which the Agent belongs
    3. Household Cluster - an index corresponding to the Agent's household cluster
    4. Working Group - an index corresponding to the Agent's work group (zeros indicate that the Agent does not belong to a working group, and negative indices indicate school mixing groups).
    5. SA1 - the region at Statistical Area Level 1 in which the Agent lives.

- WG.dat* : a $3 \times N_{DZN}$ matrix in which each row represents a destination zone partition (DZN) and the columns correspond to the following:

  1. The 11-digit DZN code corresponding to a unique employment region.
  2. The number of working groups in all previously listed DZNs (i.e., the value for the first item in the list is zero).
  3. The number of working groups located in the corresponding DZN.

  Since the working groups are indexed as they are generated, this list allows specification of the working group indices located in each DZN region.

- CD_DZ_Nstudents.dat : a list of student commuter numbers between each SA1 $\rightarrow$ DZN pair. The columns correspond to:

  1. The 7-digit SA1 code.
  2. The 11-digit DZN code.
  3. The number of students residing in the corresponding SA1 and going to school in the corresponding DZN.

  This data set is redundant in that it can be reproduced by cross-referencing population.dat and WG.dat. In combination with the TravelToWork.dat input file, it gives an estimate of commuting numbers between each SA1 and DZN location.

- SLA_Age_output.dat : This is a matrix where each row corresponds to a unique SA2 partition and the columns correspond to the following:

  1. The 5-digit SA2 identifier.
  2. Columns 2 - 6 correspond to the number of individuals from each of the five age groups, residing in the corresponding SA2.

- SCHOOLinDZ.dat* : a matrix where each row corresponds to a different school and the columns correspond to the following:

  1. The school's index.
  2. The DZN in which the school is located.
  3. The number of students attending the school.

  This dataset allows reconstruction of the spatial distribution of schools, conditional on their student populations.

  *these data files are used by the epidemic simulator in the final implementation of the population data structure. The file names not marked with an (*) are only produced for validation and visualisation purposes.

III. PART B: EPIDEMIC SIMULATOR

This section is divided into two parts. The first will describe how to run simulations of COVID-19 pandemics with no interventions. The second will describe how to apply social distancing interventions as described in the article by Chang *et al.* "Modelling transmission and control of the COVID-19 pandemic in Australia" (2020).

A. Basic epidemic simulator

The AMTraC-19 simulator as described in Chang *et al.* (2020) is implemented as follows.

1. *compilation*

To compile the AMTraC-19 program, navigate to the directory containing the source files and execute the commands in the bash script called 're_compile'. This will use GNU Autotools to configure and make the program. Makefile.am contains the compiler flags and lists headers and .cpp source files while configure.ac contains generic configuration commands. This should produce an executable called "AmTraC.exe".

2. *input arguments and files*

Table I lists the basic input arguments, their flags, and default values.

argument	flag	default value	description
trans_scaler	-t	3.33	modulates transmission probability (3.33 corresponds to $R_0 \approx 2$ for 2006, 2011, and 2016)
n_runs	-r	1	number of instances
n_days	-d	196	number of days for the simulation
database_output	-b	./AmTraC_outputs/ 2016_population_1/	directory into which AmTraC will write (note: this directory must contain a sub-directory called 'results')
database_input	-f	./AmTraC_inputs/ 2016_population_1/	directory from which AMTraC-19 will read input
reuse_seed	-u	0 (false)	(bool) flag telling AMTraC-19 to use a specific set of random number seeds read from the file database_input /'loaded_seed.dat'

TABLE I. Input arguments, their command line flags, and a brief description of the parameters.

Some details: The database_input and database_output strings are likely to be different for each implementation, remember to create a folder called 'results' within the database_output directory. The file 'loaded_seed.dat' should contain a column of at least n_runs integer values (they will be read out from the list sequentially as each run begins).

The value trans_scalar independently controls R_0 (as described in Cliff *et al.* "Investigating Spatiotemporal Dynamics and Synchrony of Influenza Epidemics in Australia: An Agent-Based Modelling Approach" (2018)) and we found that this dependence was consistent across the 2006, 2011, and 2016 populations, however, the relationship is expected to depend on population structure/demographics and may vary in other situations. Calibration can be carried out by initialising many independent index cases and locating the mean number of secondary cases produced for each value of trans_scalar tested. An updated description of R_0 calibration is included in the manuscript by Zachreson *et al.* "Interfering with influenza: nonlinear coupling of reactive and static mitigation strategies" (2020), which includes Age-stratified biases for a more accurate calibration of R_0.

The following input files are necessary for the program to run:

- population.dat*

- WG.dat*

- CDinSLA.dat*

- DZinSLA.dat*

- SCHOOLinDZ.dat*

- SLA.dat : a two-column data file, the first column contains the list of 5-digit SA2 codes, and the second column contains a dummy index list of increasing integers.

- SeedSLAs.dat: a data file containing information relevant to continuous seeding of index cases: a matrix with rows equal to the number of international airports into which index cases can be introduced. The first column is the daily incoming international air traffic, and the subsequent columns list the 5-digit SA2 identifiers for SA2s within infection range of the corresponding airport (i.e., the airport corresponding to the row). The probability of a new index case being introduced is then a function of the incoming traffic and a scaler value 'expected_number_infected_' which is defined in the file 'epidemic.cpp' (the default is 0.00004, the proportion of infections per incoming traveller). This format can be used for a broad range of seeding conditions that do not have to correspond to airport traffic. The first entry in a row simply corresponds to the relative incoming infection strength, while the subsequent entries are the SA2 regions affected by that infection strength. The probability of infection for each individual residing in the SA2 regions listed in each row is equal to

$$p_{index} \;=\; [\text{incoming passengers (row)}] \;\times\; [4\times10^{-5}] \;\times\; [\text{n affected individuals}]^{-1}\,, \tag{1}$$

  where the last term is the total number of individuals living in all SA2s listed in the corresponding row. On average, each incoming 'passenger' has a 0.004 % chance of producing an index case.

*produced by the generate_population.cpp script.

3. command line output

The simulation will produce command line output to monitor the progression of each run. This provides useful information on the number of index cases, incidence, prevalence etc.. [Note: if a segmentation fault is produced just after the program initialises, double-check that all required input files are located in the input directory.]

4. output files

The simulation will create two layers of output directories, one for the entire set of runs, and a subdirectory for each individual run. The names of these directories contain important information about the parameters used (parameter information will also appear in the command line output):

The upper-level directory name is formatted as: T[trans_scaler]_D[date]_[time]_AP

The initials _AP stand for 'airport' and indicate the type of seeding algorithm used for introducing index cases, airport seeding is the default for the variable seed_type which controls how seeding is implemented. This can be altered, but at this point only two other options have been implemented. These are (a) NumberOfSLAs and (b) PopulationRandom, which implement random seeding protocols and require the additional command line flags with specified values.

These will not be discussed here as the seeding protocols were specific to a particular investigation by Cliff *et al.*(2018).

The subdirectory names for each run are formatted as: R[run index]_[date]_[time]

Each simulation will produce the following output files:

- I_SLA.dat, a matrix of timeseries incidence data, with one column for each SA2 region (the row indices correspond to simulation each day of the simulation).

- I_SLA-[0-4].dat, each file corresponds to the location-specific incidence timeseries for each of the five age groups.

- I_detection_Ag.dat, a matrix of timeseries incidence (detection) data, with one column for each age group (the row indices correspond to each day of the simulation).

- I_illness_Ag.dat, a matrix of timeseries incidence (illness) data, with one column for each age group (the row indices correspond to each day of the simulation).

- I_infection_Ag.dat, a matrix of timeseries incidence (infection) data, with one column for each age group (the row indices correspond to each day of the simulation).

- P_SLA.dat, as with I_SLA.dat, but tabulating prevalence rather than incidence.

- P_SLA-[0-4].dat, location-specific prevalence timeseries for each age group.

- P_detection_Ag.dat, as with I_detection_Ag, but tabulating prevalence (detection) rather than incidence (detection).

- P_illness_Ag.dat, as with I_detection_Ag, but tabulating prevalence (illness) rather than incidence (illness).

- P_infection_Ag.dat, as with I_detection_Ag, but tabulating prevalence (infection) rather than incidence (infection).

- IP.dat, the incidence timeseries (first column) and prevalence (second column) for the whole population.

- Iinfected_Pinfected_CDI_CDP.dat - a four-column data set. The first column is the incidence of infected individuals (including asymptomatic cases), the second column is the prevalence of infected individuals (including asymptomatic cases), the third column is the incidence of SA1 regions transitioning from 0 infected residents to > 0 infected residents, and the fourth column is the number of SA1 regions with > 0 infected residents, the row numbers correspond to time in days.

B. Social distancing interventions

Pandemic interventions have been included in the program, as described in Chang *et al.* (2020). These include randomly-assigned social distancing, case isolation, household quarantine, and school closure.

1. *input arguments and files*

In addition to the input files listed for the basic implementation, application of social distancing interventions requires the following:
The input file 'SD_intervention_input.dat', which must be located in the input directory specified by the command line argument with flag -f. This file will specify the intervention-specific parameters. Follow the format contained

in the existing parameter file, the order of parameter specification is not important. The free parameters associated with social distancing interventions are:

- SD_intervention: boolean flag for turning on social distancing [0, 1]
- n_ill_SD_trigger: cumulative number of illnesses triggering social distancing]
- compliance_rate_SD: proportion of population who comply with social distancing, range [0, 1]
- duration_SD: number of days social distancing measures apply after they are implemented
- FoI_SD_community: multiplier for community interactions under social distancing, range $[0, +\infty]$
- FoI_SD_work: multiplier for workplace interactions under social distancing, range $[0, +\infty]$
- FoI_SD_home: multiplier for household interactions under social distancing, range $[0, +\infty]$

- CI_intervention: boolean flag for turning on case isolation [0, 1]
- compliance_rate_CI: proportion of population who comply with case isolation, range [0, 1]
- FoI_CI_community: multiplier for community interactions under case isolation, range $[0, +\infty]$
- FoI_CI_work: multiplier for workplace interactions under case isolation, range $[0, +\infty]$
- FoI_CI_home: multiplier for household interactions under case isolation, range $[0, +\infty]$

- HQ_intervention: boolean flag for turning on home quarantine [0, 1]
- compliance_rate_HQ: proportion of population who comply with home quarantine, range [0, 1] (NOTE: home quarantine compliance is conditional on case isolation compliance.)
- FoI_HQ_community: multiplier for community interactions under home quarantine, range $[0, +\infty]$
- FoI_HQ_work: multiplier for workplace interactions under home quarantine, range $[0, +\infty]$
- FoI_HQ_home: multiplier for household interactions under home quarantine, range $[0, +\infty]$

- School_closure: boolean flag for turning on school closure [0, 1]
- n_ill_SC_trigger: cumulative number of illnesses triggering school closure]
- duration_SC: number of days school closure measures apply after they are implemented
- FoI_SC_community: multiplier for community interactions under school closure, range $[0, +\infty]$
- FoI_SC_work: multiplier for workplace interactions under school closure, range $[0, +\infty]$
- FoI_SC_home: multiplier for household interactions under school closure, range $[0, +\infty]$
- p_parent_stays_home: probability that a parent will stay home under school closure, range [0, 1].

In order for AMTraC-19 to read the parameter inputs listed in 'SD_intervention_input.dat', the commandline argument:

```
-q <name of parameter input file>
```

must be entered at runtime. The default argument is "SD_intervention_input.dat", as mentioned above, this file must be located in the database input directory specified by the command line argument with flag -f.

2. *command line output*

The command line output will display the numbers of individuals who are affected by each of the social distancing measures. It will also print the input parameters at the beginning of each run. [Note: if a segmentation fault is produced just after the program initialises, double-check that all required input files are located in the input directory.]

3. *output files*

In addition to the output files listed for the basic implementation of the program, the following additional output file will be created:

- CI_HQ.dat - a two-column data set. The first column is the number of individuals affected by case isolation, and the second column is the number of individuals affected by home quarantine.

C. Vaccination intervention

Pharmaceutical interventions (i.e., vaccination campaigns) have been included by allocating two types vaccines (priority and general), using predefined age-dependent vaccine allocation ratios, as described in Zachreson *et al.* (2021). Two types of vaccination campaigns are implemented: pre-pandemic coverage and progressive vaccination rollout.

1. *input arguments and files*

Simulation of vaccination rollout requires the input file "pharm_intervention_input.dat", which must be located in the input directory specified by the commend line argument with flag -f. This file will specify the vaccination-specific parameters as follows:

- Pharm_intervention: Boolean flag for turning on vaccination [0, 1].
- n_vac_pre_epidemic_1: the number of individuals receiving the priority vaccine in a pre-pandemic mode.
- n_vac_per_week_1: the number of individuals receiving the priority vaccine per week in a progressive rollout mode.
- n_vac_pre_epidemic_2: the number of individuals receiving the general vaccine in a pre-pandemic mode.
- n_vac_per_week_2: the number of individuals receiving the general vaccine per week in a progressive rollout mode.
- p_vac_65_and_over: the number of vaccines allocated to older adults aged over 65, in each iteration of the age-stratified allocation cycle
- p_vac_18_to_64: the number of vaccines allocated to adults aged between 18 and 64, in each iteration of the age-stratified allocation cycle
- p_vac_children: the number of vaccines allocated to children aged under 18, in each iteration of the age-stratified allocation cycle
- VEs_1: the efficacy for susceptibility for the priority vaccine
- VEi_1: the efficacy against infectiousness for the priority vaccine
- VEd_1: the efficacy for disease for the priority vaccine
- VEs_2: the efficacy for susceptibility for the general vaccine
- VEi_2: the efficacy against infectiousness for the general vaccine
- VEd_2: the efficacy for disease for the general vaccine

In order for AMTraC-19 to read the parameter inputs listed in 'pharm_intervention_input.dat', the following command line argument must be entered at runtime:
-V <name of parameter input file>
The default argument is "pharm_intervention_input.dat".

2. *command line output*

The command line output will display the discrete-step (i.e., for each simulation day) and the cumulative (i.e., total) number of vaccinated individuals by age groups (i.e., in accordance with the vaccine allocation ratio) and the type of vaccine received (i.e., priority or general). It will also print out the input parameters at the beginning of each run.

3. *output files*

In addition to the output files listed in section III.A.4, the following additional output files are created:

- I_detection_VACge_Ag.dat, a matrix of timeseries incidence (detection) data for the individuals who have received the general vaccine, with one column for each age group (the row indices correspond to each day of the simulation).

- I_detection_VACpr_Ag.dat, a matrix of timeseries incidence (detection) data for the individuals who have received the priority vaccine, with one column for each age group (the row indices correspond to each day of the simulation).

- I_infection_VACge_Ag.dat, a matrix of timeseries incidence (infection) data for the individuals who have received the general vaccine, with one column for each age group (the row indices correspond to each day of the simulation).

- I_infection_VACpr_Ag.dat, a matrix of timeseries incidence (infection) data for the individuals who have received the priority vaccine, with one column for each age group (the row indices correspond to each day of the simulation).

- IinfGE_IinfPR_IdetGE_IdetPR.dat - a four-column data set. The first column is the incidence of *infected* individuals (including asymptomatic cases) for those who have received the *general* vaccine, the second column is the incidence of *infected* individuals (including asymptomatic cases) for those who have received the *priority* vaccine, the third column is the incidence of *detected* individuals (including asymptomatic cases) for those who have received the *general* vaccine, and the fourth column is the incidence of *detected* individuals (including asymptomatic cases) for those who have received the *priority* vaccine. The row numbers correspond to time in days.

- Vaccinations_general_Ag.dat - a four-column data set recording the number of individuals that received the general vaccine by age groups. The first column is the total number of individuals, the second column is the number of children (aged < 18), the third column is the number of adults aged $18 - 64$, the forth column is the number older adults (aged > 65). The row numbers correspond to time in days.

- Vaccinations_priority_Ag.dat- a four-column data set recording the number of individuals that received the priority vaccine by age groups. The first column is the total number of individuals, the second column is the number of children (aged < 18), the third column is the number of adults aged $18 - 64$, the forth column is the number older adults (aged > 65). The row numbers correspond to time in days.