Covid19 Indoor Transmission Model

Release 0.1.0

David Plummer

Jul 25, 2020

CONTENTS:

Infectious diseases are transmitted from an infectious person to a susceptible person through many mechanisms, including direct contact, exchange of fluids, respiritory droplets and formites. The Covid19 indoor transmission model considers transmission of the SARS-Cov-2 virus through respiritory droplets which have become aerosolised and spread through an indoor environment. The model makes some broad assumptions, the most important of which are:

- Covid19 can be transmitted between people in aerosolised respiritory droplets.
- Droplets are quickly transported throughout the entire indoor environment such that their is an equal concentration throughout the space.

CHAPTER

ONE

MICROENVIRONMENT SIMULATION

1.1 Simulation module

Python library to model the spread of infectious diseases within a microenvironment

class Simulation.**Simulation**(*simulation_name=None*, *simulation_run=None*, *microenvironment=None*, *periods=None*)

Bases: object

Class to implement a simulation using simpy discreate event simulation

The simulation class is the main controlling class for the simulation and is responsible for:

- Creating the simpy environment
- Creating the data collection environment
- Creating individual microenvironments
- Generating people
- Defining the routing for each person
- Starting and stopping the model

create_activities(*microenvironment_name*)
Create a dictionary of activities.

create_microenvironments()
Create the microenvironments used within the simulation.

create_network_routing()
Create a simple network routing.

From 'start' to 'end' via 'visit environment

create_people(*arrivals_per_hour*, *max_arrivals=None*, *quanta_emission_rate=None*, *inhalation_rate=None*)
Create a method of generating people

get_counter(*data_set_name*)
Return stored value of a counter

Parameters **{string} -- Name of the counter to get**(*data_set_name*) –

Returns number – value of the counter

get_list_of_reports()
Get the list of reports.

Returns list of strings – List of reports.

get_results(*data_set_name*)
Return stored report as a pandas dataFrame.

Parameters **{string} -- Name of the dataset to get**(*data_set_name*) –

Returns pandas dataFrame – dataFrame containing the results

run(*arrivals_per_hour=None*, *quanta_emission_rate=None*, *inhalation_rate=None*, *max_arrivals=None*, *report_time=None*)
Run the simulation

Keyword arguments: periods Number of periods to run the simulation report_time When True the simulation prints the time taken to execute the simulation to console.

1.2 Person module

Python library to model the spread of infectious diseases within a microenvironment

class Person.**Person**(*simulation_params*, *starting_node_id*, *infection_status_label=None*, *quanta_emission_rate=None*, *inhalation_rate=None*, *person_type=None*)
Bases: HealthDES.PersonBase.Person_base

Class to implement a person as a simpy discreate event simulation

The person will have various characteristics which influences the simulation

The person will have a flow around the simulation implemented as a list of activities which are callled as each one completes.

expose_person_to_quanta(*quanta_concentration*)
Calculate the amount of quanta the person is exposed to

Parameters **quanta_concentration** (*number*) – The concentration of infectious material in the environment in quanta

get_quanta_emission_rate()
Get the parsons quanta emmission rate

Returns Number – Quanta emission rate

infection_risk()
Determine risk that a patient is infected

infection_risk_instant(*quanta_concentration*)
Return the probability the person will become infections

Parameters **quanta_concentration** (*number*) – Concentration of infectious material in the environment

Returns Probability that the person will become infected.

Return type number

log_infection()
Log visitor activity within the process visitor process

log_infection_risk()
Log visitors infection risk

1.3 Microenvironment module

Python library to model the spread of infectious diseases within a microenvironment

class Microenvironment.**Microenvironment**(*simulation_params*, *environment_name*, *volume*, *air_exchange_rate*, *capacity=None*)

Bases: object

Class to implement a microenvironment as a simpy discreate event simulation

add_quanta_to_microenvironment(*quanta*)
Callback from person class to add quanta to the microenvironment

Arguments: quanta The number of quanta to add to the microenvironment

get_active_users()
Get the number of people in the microenvironment

Returns {integer} – Number of active people in the microenvironment

get_quanta_concentration()
Callback from person class to get the quanta concentration

get_queue_length()
Get the number of people waiting in the queue

Returns {integer} – Number of people waiting in the queue

initialise_periodic_reporting()
Initialise periodic reporting

periodic_reporting_callback()
Callback to collect data for periodic reporting

request_entry()
Request entery into the microenvironment

Returns simpy resource – Simpy resource (with potential capacity constraint)

run()
Calculate the new quanta concentration in the building

1.4 DiseaseProgression module

Python library to model the spread of infectious diseases within a microenvironment

class DiseaseProgression.**DiseaseProgression**(*infection_status_label=None*)

Bases: object

Disease status of person within the model

disease_states = ['susceptible', 'exposed', 'infected', 'recovered']

is_state(*infection_status_label*)
Tests disease state and return True if matches

set_state(*infection_status_label*)
Sets the disease state

static valid_state(*infection_status_label*)
Checks text and returns text if it is a valid disease state

CHAPTER

TWO

HEALTHDES DISCRETE EVENT SIMULATION TOOLKIT

The HealthDES framework provides a set of base class modules to model health and care processes using discrete event simulation. The framework supports the definition of people, resources that they need and activities the people engage in as they transition a network graph. The network graph defines the ordering of activities and the resources people need at each point as they traverse the care pathway.

2.1 PersonBase module

2.2 MicroenvironmentBase module

2.3 Activity module

Python library to model the spread of infectious diseases within a microenvironment

class Activity.**Visitor_activity**(*simulation_params*, ***kwargs*)

Bases: object

Person's activity within the system, models interaction between people and environment

infected_visitor(*callback_add_quanta*, *request_to_leave*, *periods*)

Callback from microenvironment for an infected person to generate quanta

Arguments: callback_add_quanta Callback to microenvironment to add quanta request_to_leave Event notification to let microenvironment know we wish to leave

to allow clean up before existing the microenvironment.

periods Number of periods person in the microenvironment

log_visitor_activity(*activity*)

Log visitor activity within the process visitor process

Arguments: activity String describing the activity that has occurred.

classmethod pack_parameters(*microenvironment*, *duration*)

Pack parameters for the activity into a dictionary.

Parameters

- **{microenvironment obj} -- Microenvironment that the person will enter**(*microenvironment*) –
- **{number} -- Amount of time person spends in the environment**(*duration*) –

Returns Tuple(class, dictionary) – This class and a dictionary of parameters required to instantiate an instance

start (*finished_activity*)
Introduce a person to the microenvironment

Arguments: finished_activity Event notification that activity has completed duration Length of time that infected person remains in the microenvironment

susceptible_visitor (*callback_quanta_concentration*, *request_to_leave*, *periods*)
Callback from microenvironment for a susceptible person to calculate exposure

Arguments: callback_quanta_concentration callback to microenvironment to get quanta_concentration request_to_leave Event notification to let microenvironment know we wish to leave

to allow clean up before existing the microenvironment.

periods Number of period that person in the microenvironment

unpack_parameters (***kwargs*)
Unpack the parameter list and store in local instance variables.

2.4 Routing module

2.5 DataCollection module

2.6 Check module

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

S

U

V