Additional file 1 – Algorithms

Table of contents

Algorithms used for this paper .................................................................1
Description of symbols used in this document .........................................2
Contact generation for the random case ..................................................3
Contact generation for the repetitive case ...............................................4
SIR model ..............................................................................................7

Algorithms used for this paper

We describe in this document the central algorithms on which the simulations of this paper base. The algorithms are given as flowcharts. These flowcharts are designed following the international standard ISO 5807-1985. The symbols used in this document are briefly described below. Detailed descriptions can be found in the ISO 5807-1985 documentation.

Where necessary, subparts of the algorithms are described in detail. Straightforward operations are not described in detail but as generic processes.

In this supplementary material we describe the following algorithms:

- Generation of contacts in the random mixing case
- Generation of contacts in the repetitive contacts case (with clustering)
- SIR model

Algorithms used for other kinds of data handling, data presentation or storage and analysis are not presented.

The core of our simulation software is structured into two main processes: (1) the generation of contacts and (2) the simulation of disease spread.

The contact generation algorithms generates for each individual in the population pre-defined sets of \( n \) contact partners for \( \tau \) different days, whereas \( n \) is the number of contacts per day and \( \tau \) is the infectious period. There exist two version of the contact generation algorithm: one for the random mixing case and one for the case of repetitive contacts allowing for clustering. The generated contacts are stored in a contact list for each individual.

The algorithm for the disease spread makes use of these pre-defined contact list by simulating stochastically how a disease with given parameters spreads through such a given structure.

---

1 ISO 5807-1985: Information processing – Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts.
Description of symbols used in this document

**Process:**
Used for any kind of defined operation or group of operations.

**Data:**
Used as representation of data

**Stored data:**
Used as representation of stored data

**Begin of loop:**
1\textsuperscript{st} line indicates either a WHILE loop or a counter;
2\textsuperscript{nd} line gives termination criterion

**End of loop:**
Go to begin of loop if termination criterion is not fulfilled

**Decision:**
Depending on whether the given statement is true or false, the one or the other path is used.

**Predefined process:**
Used for a defined process specified elsewhere in this document

**Terminator of a predefined process**
Contact generation for the random case

Main routine for the generation of non-repetitive contacts:

- **person** loop over all individuals
  - **position_1** loop over all contacts to be generated
    - **flag** = true
      - **WHILE** flag = true
        - Draw one individual randomly from the population
        - **position_2** loop over all contacts already generated
          - **flag** = false
            - **loop** over all contacts already generated
              - Store drawn individual as contact at **position_1**
        - **END WHILE** terminate if **flag** = false
    - **END WHILE**
      - **position_2** loop over all contacts already generated
        - Store drawn individual as contact at **position_1**

- **person** end loop
  - **position_1** end loop
Contact generation for the repetitive case (allowing for clustering)

Main routine for the generation of repetitive contacts:

```
WHILE contact_list of at least two persons not entirely filled

    agentlist

    random contacts

    WHILE number of contacts added in last loop > 0

        agentlist

        cluster contacts

    END WHILE

    terminate if contact_list full

END WHILE

terminate if no contact added
```

Subroutine “agentlist”:

```
SUBROUTINE agentlist

    person loop over all persons

    contact_list of person full

    Store current person in list of agents needing further contacts

    person end loop

END SUBROUTINE
```
Subroutine „random contacts“:

SUBROUTINE random contacts

WHILE agents left in agentlist

Draw person 1 randomly from agentlist

Draw person 2 randomly from agentlist

loop

end loop; no termination criterion

person 1 = person 2

false

true

Contact between person 1 and person 2 exists already

false

true

Pick next individual in agentlist as person 2

loop

end loop

Store person 1 in contact list of person 2 and vice versa

Delete person 1 and person 2 from agentlist

END WHILE

terminate if agentlist empty

END SUBROUTINE
Subroutine „cluster contacts“:

```
SUBROUTINE cluster contacts
WHILE agents left in agentlist
  WHILE agents left in agentlist
    Draw one person randomly from agentlist
    contact 1 loop over all contacts of the drawn person
      contact_list_of contact 1 full
        contact 2 loop over all contacts of the drawn person
          contact_list_of contact 2 full
            contact 1 = contact 2
              false
              contact between contact 1 and contact 2 established
                Count new triplets induced by a contact between contact 1 & 2
                true
                Store contact 1 in contact list of contact 2 and vice versa
                false
            false
          false
    end loop
    true
  end loop
end loop
true
END WHILE terminate if agentlist empty
END SUBROUTINE
```

Current # triplets ≥ triplet aim
true
false

Contact list of contact 1 full
true
false

Contact list of contact 2 full
true
false

Contact 1 = Contact 2
true
false

Contact between contact 1 and contact 2 established
true
false

Count new triplets induced by a contact between contact 1 & 2
true
false

Store contact 1 in contact list of contact 2 and vice versa
true
false

End loop

SIR model

Main routine for the simulation of disease spread:

WHILE # infected ≠ 0

Increase week-day by one. If weekday = 15 set it back to 1.

person loop over all persons

status(person) = infected

infect others

Reduce counter for remaining infectious days by one

Counter for remaining infectious days = 0

true

status(person) = recovered

false

person loop over all persons

Person infected in subroutine "infect others"?

true

Set up counter for infect. days. Status(person) = infected

false

person end loop

END WHILE

terminate if # infected = 0

person end loop
Subroutine „infect others“:

SUBROUTINE infect others

contact
loop over all contacts of person & weekday

status(person)=suscep.
true
false

Draw random number with \(0 \leq x < 1\) (uniformly distributed)

random number < infection probability
true
false

Store contact partner as to be infected in next time step

contact
end loop

END SUBROUTINE