CHARM Documentation

V3.0 – 11.10.2022

Climate HARM This microsimulation models multiple health outcomes as a result of changing climate exposures. Climate projections are given to the model as inputs, along with individual non-climate characteristics for the population, and health outcomes of interest are estimated for a defined time period. The main structure of the model is based on Krijkamp et al (2018)1 and is described [here](algorithms.docx).

The model consists of several separate files:

* **MicroSim:** This contains the MicroSim() function (i.e., the loops taking each individual through each time cycle).
* **Probs:** This contains the Probs() function, which is used to assign health states to individual at each consecutive time cycle, given their current health state and individual characteristics and exposures.
* **Costs:** This contains the Costs() function, which is used to assign costs to each individual in each cycle, given their health state.
* **Effs:** This contains the Effs() function, which is used to calculate quality-adjusted life years for each individual in each cycle.
* **modifyRisk:** Used to update the vector of transition probabilities given individual risk factors in the current cycle.

# MicroSim

This is the main microsimulation function containing the model. The MicroSim function accepts the following arguments:

* initStates: the vector of initial health states of individuals at initiation
* n.i: total number of individuals at initiation
* n.t: total number of cycles to go through
* m.fire: fire data
* v.n\_asthma: vector of asthma health state names
* m.x: microdata; matrix with individuals’ data
* cl: cycle length
* birthRate\_bl: annual birth rate in the population at baseline
* birthRate\_change: annual change in birth rate
* allCauseMortality\_change: annual change in all cause mortality
* d.c: discount rate for costs
* d.e: discount rate for health state utilities
* intervention: intervention coverage/reach (required argument; set to 0 if no intervention)
* intervention\_trigger: the level of exposure that triggers deployment of intervention
* seed: starting seed number, default is 1
* debug: Boolean to run the model with or without debugging outputs, default is FALSE

## Initialization

The first chunk of code initializes vectors and matrices that will be populated as the simulation loops through each cycle.

Two vectors are created to hold discount weight for costs (v.dwc) and QALYs (v.dwe) in each cycle.

The vector v.birthRate is calculated with birth rates in each cycle, and the vector v.deathRate is calculated holding the death rate adjustment value in each cycle (not the actual death rate, as this value is taken from the probability matrix).

Matrices m.I, m.T, m.M, m.C, and m.E are initialized to store the intervention, therapy, health state, costs, and QALYs of each individual in each cycle, respectively. These matrices have dimensions n.i x (n.t+1); each row represents an individual and the columns show their status at each cycle, starting with “cycle 0” which is the initial status in which they enter the simulation.

A treatment tracker (txTracker) is initialized to track asthma therapy history of individuals as they move through the simulation. This matrix has a row for all possible combinations of individual-therapy pairs (e.g., if there are 7 possible therapies including no therapy, there should be 7 rows for each individual), and columns to store when the person started the latest round of a specific therapy (txStart), when that latest round ended (txEnd), whether the latest round was successful (txSuccess), and a lifetime count of completing that therapy (txCount). For details, see Section ### of this document.

A vector of length n.t+1, v.totalPop is created to track population size over each cycle, with the first element being n.i.

## Simulation

### Beginning of a cycle

At the start of each cycle, the variable totalBirths\_t is set (or reset) to 0, to keep count of all the births within that cycle. Then, **the model loops through each individual one by one**.

### Looping through individuals

#### Climate change exposure and intervention

The model first pulls the individual’s geographic location and determines whether a fire happened in that area and whether they were exposed to a fire at a nearby location (the extent of this exposure effect depends on the distance from the fire). Using this information, the model decides whether the individual will receive the intervention to mitigate the impact of a fire (table ##). Note that this is not an asthma therapy, which are modeled separately and called “therapies”; an “intervention” is a non-clinical intervention to mitigate the impact of wildfire exposure on asthma. If an intervention is being modeled and the individual was exposed to a fire, they are eligible for the intervention in that cycle. The probability of an eligible individual receiving the intervention is equal to intervention coverage. The intervention assignment is saved into m.I.

Table . Assignment of interventions to individuals

|  |  |  |  |
| --- | --- | --- | --- |
| **IF:** | | | THEN |
| Intervention coverage > 0 | AND | Fire in neighborhood == 1  OR  Exposure to a fire nearby >= predetermined exposure level that triggers an intervention | P(receive intervention) = intervention coverage |
| **ELSE:** | | | No intervention |

#### Calculation of the new health status

The next chunk of code recalculates and assigns the individual’s health status for the next cycle. It first calculates an updated vector of transition probabilities for the individual using their individual risk factors and current health status, then it samples a new health status using these updates probabilities. This is done by the Probs() function. Briefly, this function takes current health state, fire exposure, individual risk factor data, intervention status, and the death rate adjustment factor for the current cycle as arguments. It then pulls the relevant row from the transition probability matrix, recalculates each probability in that row for the individual, and outputs a vector containing adjusted probabilities for all possible health statuses the individual can transition to. The next health status is then sampled randomly given this adjusted probability vector, and saved into m.M.

#### Revising asthma therapy

For asthma, the algorithm for evaluating therapies is as follows:

In each cycle, if the individual has **poorer control *or* has been on the same therapy for a pre-determined number of cycles**, their therapy is reevaluated given their asthma control status during that cycle and the probability of receiving a certain therapy based on asthma control status. The txTracker matrix is updated accordingly. If neither of these conditions are true, they remain on the same therapy as the previous cycle. Whenever an individual gets a therapy reevaluation, the therapy they stop is also evaluated to determine whether it was successful. If the individual **maintained their control status from the previous cycle *or* had poorer control** (i.e., they did not get better), the therapy is recorded as a failure. Otherwise (if they do get better), the therapy is recorded as a success.

*Example:*

In a model where therapies are reviewed and updated every 2 cycles, the txTracker for an individual (id=78) may look like the following at the end of cycle 8:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **tx** | **txStart0** | **txEnd** | **txSuccess** | **txCount** |
| 78 | S1 | 0 | 1 | 0 | 1 |
| 78 | S2 | 7 | 7 | 0 | 2 |
| 78 | S3 | 3 | 5 | 1 | 1 |
| 78 | S4 | NA | NA | NA | 0 |
| 78 | S5 | NA | NA | NA | 0 |

From this table, it is possible to see that the individual was on treatment S1 at initiation and switched to a different therapy in cycle 1 (as S1 was not successful). The therapy they switched to is not visible since the tracker only records data for the most recent round of each therapy, however this information is stored in the matrix m.T if needed. In this case, it can also be inferred from the data available in the tracker. Since S1 failed, and S3 was only completed once, this individual was on S2 between cycles 1-3, which was not successful. In cycle 3, they stepped up to S3 therapy and improved their disease status. This allowed them to step back down to S2 in cycle 5 (therefore completing two full rounds of S2 between 1-3 and 5-7). They remained in S2 in cycle 7 but have not completed that round since the simulation ended at cycle 8 before the therapy was reviewed.

#### Calculation of costs and QALYs

The costs incurred within that cycle are calculated and saved in matrix m.C using the Costs() function. The QALYs for that individual during that cycle are calculated and saved in matrix m.E using the Effs() function.

#### Updating the cohort for the next cycle

The cohort is aged by adding the cycle length to the age variable of each individual. If a fire exposure occurred, the exposure history is set to TRUE.

The next chunk of code is only relevant for open cohort models. It calculates births, generates the new individuals (i.e., assigns their individual characteristics and risk factors), and adds these new data to the microdata dataset m.x.

First, it determines if the individual had children from a Poisson distribution where the probability (lambda) is the birth rate at the current cycle. If there was a birth, a new row is created in matrix m.x to describe that new individual:

* A new id number is assigned.
* Age is entered as zero.
* Sex is determined from a binomial distribution.
* Fire exposure is determined based on whether a fire happened in the neighborhood in that cycle.
* Same rural/urban designation as the parent is assigned.
* The asthma care status is entered as no treatment.
* Same neighborhood as the parent is assigned.
* Same family id number as the parent is assigned.

New rows are also added the m.M, m.I, m.T, m.C, and m.E to track the individual’s information in the following cycles.

Finally, the new births are added to totalBirths\_t to calculate the total number of new individuals within that cycle. Then, **the loop for individuals is closed**.

### Ending each cycle

Once the model loops through each induvial for a cycle, it updates the population size (n.i) by adding the total number of births (totalBirths\_t). This new population size is saved in a vector that keeps track of the population size across each cycle (v.totalPop).

Total discounted costs and total discounted QALYs are calculated for each individual by multiplying matrix m.C with the vector of cost discount factor, and m.E with the vector of QALY discount factors, respectively. Mean discounted costs and mean discounted QALYs are then saved into separate variables.

Lastly, two simulation traces are created to depict the health status of the population across the simulation. These traces have asthma control statuses as rows and cycles as columns. One trace (TR.absolute) shows the absolute number of individuals occupying each health status at each cycle. The other (TR.proportion) shows the percentage of the population occupying each health status at each cycle.

### Outputs

The following outputs are returned as a list:

* m.M: matrix with the health status of each individual at each cycle
* m.I: matrix with the intervention assignment of each individual at each cycle
* m.T: matrix with the asthma therapy assignment of each individual at each cycle
* m.C: matrix with the costs incurred by each individual at each cycle
* m.E: matrix with the QALYs of each individual at each cycle
* tc: total discounted costs for each individual
* te: total discounted QALYs of each individual
* tc\_hat: average discounted costs
* te\_hat: average discounted QALYs
* m.x: revised microdataset at the end of the simulation
* TR.absolute: simulation trace with absolute number
* TR.proportion: simulation trace with proportions
* txTracker: asthma therapy tracking data
* if applicable, debugging tracker: additional process information for debugging

# Probs

This function calculates transition probabilities for each individual at each cycle given individual risk factors and previous health status. Its arguments are:

* M\_it: the health status of individual i at cycle t
* fire\_it: whether a fire exposure was experienced by individual i at cycle t
* x\_i: individual characteristics and risk factors of individual i (i.e., the ith row of microdata matrix m.x)
* intervention\_it: the intervention assignment of individual i at cycle t
* deathrate\_t: the death rate adjustment factor for cycle t
* debug: Boolean to run the model with or without debugging outputs

The function takes the current health status of the individual and pulls the corresponding row from the transition probability matrix into the vector v.p.it (vector of probabilities for individual i at cycle t). It then loops through each element in ***v.p.it*** and recalculates them based on current risk factors and interventions. For each non-dead health status an individual can transition into, two vectors are created:

* ***v.x***: This holds the individual risk factors that will affect the probability of transitioning from the current state to the next. Each element represents one risk factor and must be a 1/0 Boolean.
* ***v.rr***: This vector pulls the appropriate risk ratios for each transition probability.

Importantly, these two vectors must be the same length, and the order in which the risk factors are given must follow the same order as the corresponding risk ratios.

Next, for any non-“stay-put” transition, a new probability is calculated by multiplying the base probability with an adjustor value. This adjustor value is determined via a simple function, ***modifyRisk()***, calculates the dot product of ***v.x*** and ***v.rr***.

The asthma-adjusted all-cause death rate is re-calculated outside of this loop to incorporate the change in death rate of time. Finally, the “stay-put” probability is calculated as one minus sum of all other probabilities.

The adjusted vector of transition probabilities for the individual is returned as ***v.p***.

**References**

1. Krijkamp EM, Alarid-Escudero F, Enns EA, Jalal HJ, Hunink MGM, Pechlivanoglou P. Microsimulation Modeling for Health Decision Sciences Using R: A Tutorial. *Med Decis Making*. Apr 2018;38(3):400-422. doi:10.1177/0272989X18754513