RSV_main.R

lwillem

Tue Feb 11 15:58:39 2020

```r
############################################################################
# This file is part of the RSV modelling project McMarcel.
#
# => MAIN SCRIPT TO RUN THE MODEL FOR 72 GAVI COUNTRIES
#
# Multi-Country Model Application for RSV Cost-Effectiveness poLicy (McMarcel)
#
#  Copyright 2020, CHERMID, UNIVERSITY OF ANTWERP, BELGIUM
############################################################################
# The objective of this modelling project is to evaluate the impact and cost-
# effectiveness of potential maternal and neonatal RSV immunisation strategies
# in 72 Gavi countries. See our README file for more info.
#
# Citation: Li, Willem, Antillon, Bilcke, Jit, Beutels. Health and economic
# burden of Respiratory Syncytial Virus (RSV) disease and the cost-
# effectiveness of potential interventions against RSV among children under
# 5 years in 72 Gavi-eligible countries. BMC Medicine. (2020)
############################################################################

# clear workspace
rm(list=ls())

## set working directory (or open RStudio with this script)
# setwd("C:/User/path/to/the/rcode/folder") ## WINDOWS
# setwd("/Users/path/to/the/rcode/folder") ## MAC

#######################
## SETTINGS          ##
#######################

# select the model configuration
# => set 'run_tag' to find the config file at ./config/<run_tag>.csv
run_tag  <- 'RSV_gavi72_basecase'  # 72 Gavi countries (basecase)
#run_tag <- 'RSV_gavi72_all'       # 72 Gavi countries (all scenarios)
#run_tag <- 'RSV_gavi72_efficacy'  # 72 Gavi countries (severity-specific effiacy)

# number of stochastic samples in the probabilistic sensitivity analysis (PSA)
num_sim           <- 5000

# random number generater seed
rng_seed          <- 20190118

# option to create geographical and country-specific plots
# note: this might require substantial processing time
boolean_country_plots <- TRUE
boolean_map_plots     <- TRUE
```

```r
########################
## MODEL SETUP       ##
########################

# (re)load packages and functions
source('functions/RSV_load_all.R')

# output directory postfix
output_dir_postfix        <- paste0(run_tag,'_n',num_sim)

# add timestap to output directory name
output_dir <- paste0('output/',format(Sys.time(),'%m%d%H%M%S_'),output_dir_postfix)

# set seed
set.seed(rng_seed)

# config filename
config_filename <- paste0('./config/',run_tag,'.csv')

# log timings
time_stamp_main <- Sys.time()

# always clear temporary results
cli_print('Clear all temporary output')
unlink(file.path(get_temp_output_folder(output_dir)),recursive = T)

# start parallel workers
start_parallel_workers()

cli_print("****** START MC MARCEL ******")
cli_print("WORK DIR:",system('pwd',intern = T))
cli_print("OUTPUT DIR:",output_dir)

########################
## LOAD CONFIG       ##
########################

# load config file in csv format
sim_config_matrix <- read.table(config_filename,sep=',',
                                dec='.',stringsAsFactors = F,header = T)

# set output file name prefix
sim_output_filename  <- file.path(output_dir,run_tag)

# add simulation details
sim_config_matrix$num_sim          <- num_sim
sim_config_matrix$scenario_id      <- 1:nrow(sim_config_matrix)
sim_config_matrix$rng_seed         <- rng_seed
sim_config_matrix$outputFileDir    <- get_output_folder(output_dir)

# Count number of scenarios
num_scen <- length(sim_config_matrix$scenario_id)
```

```r
##############################################
## PRE-PROCESSING: country databases        ##
##############################################
cli_print('START PRE-PROCESSING',run_tag); time_stamp <- Sys.time()

# create UN country data
create_UN_country_database(output_dir)

# pre-process WPP2017 data
load_wpp2017_databases(output_dir)

######################################
## PRE-PROCESSING: life tables      ##
######################################
## note: for a sequential run, replace %dopar% by %do%
cli_print('PRE-PROCESSING LIFE TABLES...'); time_stamp <- Sys.time()

# construct matrix with all unique [country, year] combinations
country_year_opt      <- sim_config_matrix[,c('country_iso','year')]

# add [country, 2015] to derive the reference incidence, based on Shi et al (2017)
country_year_opt      <- rbind(country_year_opt,
                               cbind(country_iso=country_year_opt$country_iso, year=2015))
# get unique combinations
country_year_opt      <- unique(country_year_opt)

# make summary matrix, including the 5-year period notation
country_period_opt    <- cbind(country_year_opt$country_iso,
                                 t(sapply(country_year_opt$year,get_year_category)))

# get life table for each [country, period] combination
par_out <- foreach(i_life=1:nrow(country_period_opt),
                   .combine='rbind',
                   .packages=all_packages,.verbose=FALSE) %dopar%
{
  # print progress
  cli_progress(i_life,nrow(country_period_opt),time_stamp)

  # life table with discounting
  generate_life_table(country_period_opt[i_life,1],
                      country_period_opt[i_life,3],
                      output_dir,
                      0.03)

  # life table without discounting
  generate_life_table(country_period_opt[i_life,1],
                      country_period_opt[i_life,3],
                      output_dir,
                      0)

  # dummy return, the results are printed to a file
  return(0)
}
```

```r
########################################
## PRE-PROCESSING: incidence          ##
########################################
## note: for a sequential run, replace %dopar% by %do%
cli_print('PRE-PROCESSING INCIDENCE...' ); time_stamp <- Sys.time()

# get unique country codes
country_opt <- data.frame(country_iso = unique(sim_config_matrix$country_iso))

# preprocess incidence data for each country
par_out <- foreach(i_country=1:nrow(country_opt),
                   .combine='rbind',
                   .packages=all_packages,
                   .verbose=FALSE) %dopar%
{
  # print progress
  cli_progress(i_country,nrow(country_opt),time_stamp)

  # get country-specific incidence data
  get_incidence(country_opt$country_iso[i_country],
                output_dir)

  # dummy return, the results are printed to a file
  return(0)
}

########################################
## PROCESSING: burden                 ##
########################################
## note: for a sequential run, replace %dopar% by %do%
cli_print('PROCESSING BURDEN [FOREACH]:',run_tag); time_stamp <- Sys.time()

# loop over each configuration
sim_output <- foreach(i_scen    = 1:num_scen,
                      .combine  = 'rbind',
                      .packages = all_packages,
                      .verbose  = FALSE) %dopar%
{
  # print progress
  cli_progress(i_scen,num_scen,time_stamp)

  # run burden function
  run_output_long <- get_burden(sim_config_matrix[i_scen,])

  # write results to file
  save(run_output_long,file=file.path(get_temp_output_folder(
    sim_config_matrix$outputFileDir[i_scen],'burden'),
    paste0('run_output_long_',i_scen,'.Rdata')))

  # dummy return, the results are printed to a fle
  return(0)
}
```

```r
################################
## POST-PROCESSING           ##
################################
cli_print('COLLECT BURDEN OUTPUT [FOREACH]:',run_tag); time_stamp <- Sys.time()

# check parallel workers
check_parallel_workers()

# loop over each scenario
sim_output <- foreach(i_scen   = 1:num_scen,
                      .combine = 'rbind',
                      .verbose = FALSE) %dopar%
{
  # print progress
  cli_progress(i_scen,num_scen,time_stamp)

  # get output name
  var_name <- load(file.path(get_temp_output_folder(sim_config_matrix$outputFileDir[i_scen],
                    'burden'),paste0('run_output_long_',i_scen,'.Rdata')))

  # load data and add scenario id
  run_output_long             <- get(var_name)
  run_output_long$scenario_id <- i_scen

  # return
  return(run_output_long)
}

# add config details to output
sim_output <- merge(sim_config_matrix,sim_output,all=T)

# sort output on scenario_id and save as RData file
sim_output <- sim_output[order(sim_output$scenario_id),]
save(sim_output,sim_output_filename,file=paste0(sim_output_filename,'.RData'))

#######################
## PLOT RESULTS      ##
#######################

# plot CEAF table overview
plot_CEAF_table(sim_output_filename)

# plot CEA results by country
if(boolean_country_plots) { plot_CEA_country_results(sim_output_filename) }

# get geographic figures (optional)
if(boolean_map_plots) { plot_maps(sim_output_filename) }

# get aggregated global and country tables
write_global_summary_tables(sim_output_filename)

# stop parallel workers
stop_parallel_workers()
```