Notes for Amy

**To add Roxygen documentation to a function in R**

1. Click your cursor to inside the name of the function.
2. Go to Code 🡪 Insert Roxygen Skeleton. Click Insert Roxygen Skeleton.
3. Provide a title.
4. Provide a description for each @param
   1. If you add/delete parameters from the function, update the @param documentation
5. Provide a description of what the function returns in @return
6. You may delete @examples.
7. Delete @export if you don’t want the user to be able to directly use this function. Leave @export for user interface functions.
8. Good Practices
   1. Generally we have one function per R script.
   2. Save the function in a script with the same name and .R extension.
   3. Place the script in the R/ folder.

**To add documentation to an S4 class**

1. Click your cursor to inside the name of the function.
2. Go to Code 🡪 Insert Roxygen Skeleton. Click Insert Roxygen Skeleton.
3. Provide a title.
4. If desired, provide additional descriptions for each @slot
5. Delete @return
6. Delete @examples
7. Leave @export
8. Add @docType class
9. Add @rdname name-class (\*see example below)
10. If the class includes the contains field in which it contains subclasses, be sure you set the classes in order, so that classes come after subclasses they contain (\*see example below)

**Example documentation for S4 class with no subclasses**

#' Class definition for space.nMx object - aka Age Specific Death Rates (ASDR) - by space

#'

#' @slot rate.years numeric; year associated with each time-specific rate

#' @slot rates data.frame; age and space specific death rates per 1 (rows) by year (columns)

#' @slot mid.age numeric; mid-age associated with each age-specific rates

#' @slot n.subpops numeric; number of subpopulations

#'

#' @export

#' @docType class

#' @rdname space.nMx-class

setClass("space.nMx",

representation(rate.years="numeric", #year associated with each time-specific rate

rates="data.frame", #age and space specific death rates per 1 (rows) by year (columns)

mid.age="numeric", #mid-age associated with each age-specific rates

n.subpops="numeric" #number of subpopulations

))

**Example documentation for S4 class with 3 subclasses**

#' Generate of object of class experiment result (sim.result)

#'

#' Class to hold SIR simulation results. Hold the results of the simulation as

#' SIR compartment values at each time step.

#'

#' @slot .Data matrix.

#' @slot s.inds numeric.

#' @slot i.inds numeric.

#' @slot r.inds numeric.

#' @slot age.class numeric.

#' @slot t numeric.

#'

#' @export

#' @docType class

#' @rdname sim.results.SIR-class

setClass("**sim.results.SIR**",

slots = list(.Data = "matrix",

s.inds = "numeric", #indexing of susceptibles

i.inds = "numeric", #infectious indexes

r.inds = "numeric", #recovered indexes

age.class = "numeric", #upper end of age classes

t = "numeric"#times when we have info in the matrix

),

contains="matrix") **# you don’t have to set the class matrix, so we start with this class**

#' Holds the results of a simulation with an MSIRV object

#'

#' @slot m.inds numeric.

#' @slot v.inds numeric.

#' @slot routine.intro numeric.

#' @slot sia.times numeric.

#'

#' @export

#' @docType class

#' @rdname sim.results.MSIRV-class

setClass("**sim.results.MSIRV**",

slots = list(m.inds = "numeric",

v.inds = "numeric",

routine.intro = "numeric",

sia.times = "numeric"),

**contains= "sim.results.SIR"**) # contains the class above

#' Holds the results of a simulation with an experiment.updatedemog object

#'

#' @slot births.each.timestep numeric.

#' @slot growth.rate.each.timestep numeric.

#'

#' @export

#' @docType class

#' @rdname sim.results.MSIRV.update.demog-class

setClass("**sim.results.MSIRV.update.demog**",

slots = list(births.each.timestep = "ANY",

growth.rate.each.timestep = "ANY"

),

**contains="sim.results.MSIRV")** # contains the class above

#' Holds the results output for a simulation with an experiment.updatedemog.vaccinationchange object

#'

#' @slot MR1.fail.each.timestep numeric.

#' @slot MR2.fail.each.timestep numeric.

#' @slot SIA.fail.each.timestep numeric.

#'

#' @export

#' @docType class

#' @rdname sim.results.MSIRV.update.demog.vaccine.change-class

setClass(**"sim.results.MSIRV.update.demog.vaccine.change",**

slots = list(MR1.fail.each.timestep = "ANY",

MR2.fail.each.timestep = "ANY",

SIA.fail.each.timestep = "ANY"

),

**contains="sim.results.MSIRV.update.demog")** # contains the class above

**Set Methods – each method type in its own script**

**For the plot methods – I just turned them into separate functions – it worked better this way.**