Proportional multi-state multiple-cohort life table model

Belen Zapata-Diomedi and Ali Abbas

25 May, 2018

Contents

1 Introduction **2**
1.1 Contribution to ITHIMR . . . 3
1.1.1 Difference between ITHIM and PMSLT . . . 3

2 R development **4**
2.1 Inputs . . . 6
2.1.1 Life table . . . 6
2.1.2 Disease life tables . . . 7
2.2 Code . . . 7
2.2.1 Set up . . . 8
2.2.2 Inputs . . . 9

3 Comments **37**
3.1 Road injuries in the PMsLT . . . 37

References **37**

1 Introduction

The proportional multi-state multiple-cohort life table model (PMSLT) is a population level model (macro) approach to simulate health (and economic) implications of changes in exposure to health risk factors (e.g. physical inactivity, air pollution and diet). The PMSLT has been widely used to simulate outcomes for population level interventions for the reduction of chronic diseases.

The model was developed by Jan Barendregt and colleagues and has been widely used in Australia and New Zealand (T. Vos et al. 2010; Blakely et al. 2015).

The basic infrastructure of the model consist of three components: (1) Effect size for the intervention of interest (e.g. intervention to urban design that modifies population levels of physical activity); (2) Calculation of the potential impact fraction (PIF) to derive the change in occurrence of disease (incidence rate/case fatality rate) attributable to a change in the distribution of the risk factor (e.g. physical activity); and (3) Use of the PMSLT to simulate health (and economic) outcomes attributable to a change in the distribution of health risk factor/s in the population of interest. Figure 1 summaries the basic infrastructure of the model. ITHIM is included in Figure 1 to show that both approaches share in common steps one and two and differ in the mechanisms of calculation of change in health burden.

HALYs, QALYs and DALYs

In this model we use the term *health-adjusted life year* (HALY). As *summary measure of population health* it measures both quantity and quality of life, where one HALY represent the equivalent of one year in full health (which could be two years with a quality of life of 0.5, for example). Specific types of HALY are the quality-adjusted life year (QALY) and the disability-adjusted life year (DALY). The QALY derives from economics and was first used in the 1960s as a measure of health gain (Gold, Stevenson, and Fryback 2002). The disability-adjusted life-year (DALY) was developed for use in burden of disease studies as a measure of health loss due to disease (Gold, Stevenson, and Fryback 2002). Our calculated HALYs are neither QALYs not DALYs, but something in between. They are similar to QALYs in that they represent health gains. However, the main difference is in the calculation of the health-related quality of life component. QALYs use measures of utility weights that traditionally represent individual experiences of health, whereas our estimated HALYs use disability weights linked to specific diseases, which were developed for the Global Burden of Disease study (Gold, Stevenson, and Fryback 2002). As discussed in past research (L. Cobiac, Vos, and Barendregt 2009; Roux, Pratt, and Tengs 2008) the main advantage of using disability weights over utility weights is that disability weights refer to specific diseases rather than health states (which are difficult to link to risk factors-e.g. physical inactivity). We opted to use the more general terms HALYs given that the use of the DALYs terminology may lead to think that our calculations are similar to those in burden of diseases studies (Murray et al. 2012). In our study, our model does not explicitly separate years of life lost (YLL) and years lived with disability (YLD) components, but instead calculates the total number of life years lived, adjusted for the average health-related quality of life in those years (by age and sex). In burden of disease studies, DALYs are defined as the sum Years of Life Lost (YLL) and Years Lived with Disability (YLD).

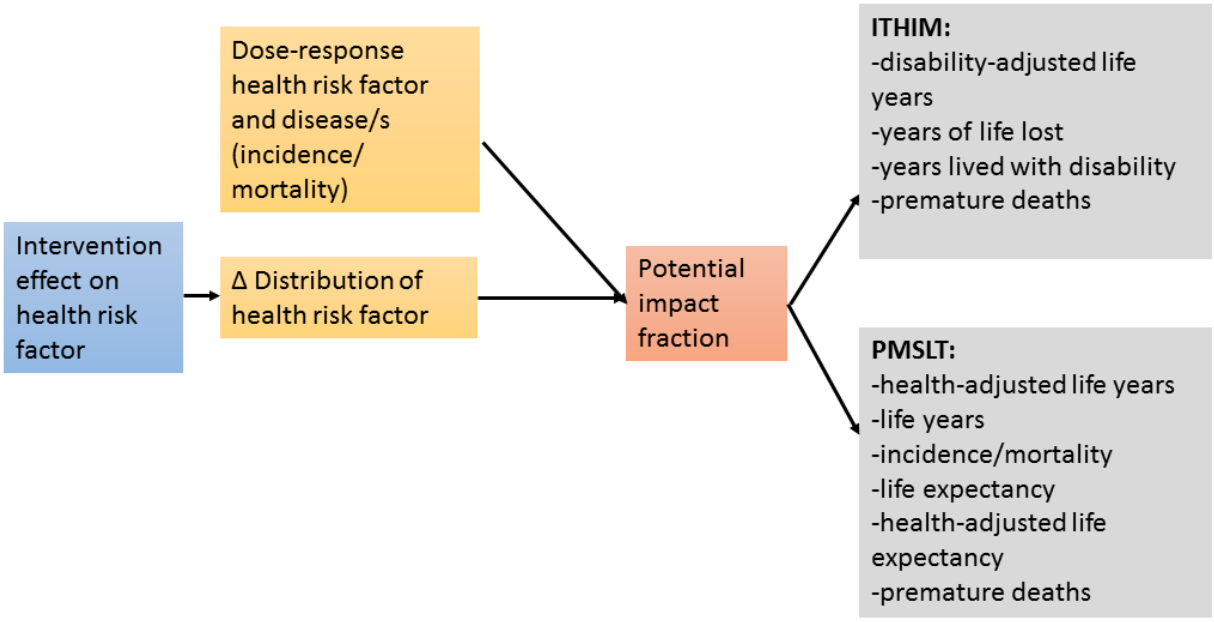

Figure 1: Basic ITHIMR infrastructure

1.1 Contribution to ITHIMR

The PMSLT similar to ITHIM is a comparative risk assessment approach (Briggs, Scarborough, and Smith 2016) that consist of calculating the change in the health burden for a population of interest from a change in exposure to health risks factors (e.g. physical inactivity, air pollution and road trauma).As depicted in Figure 1, both methods need estimates of the potential impact fraction (PIF), which indicates the proportion of the disease burden attributable to a risk factor of interest (e.g. physical inactivity) (Barendregt and Veerman 2010). A step further back, is the development of scenarios that bring about change in the distribution of the risk factor of interest. For now, we only focus on calculations from the PIF onward, and provide a hypothetical example of change in the population distribution of physical activity. Incorporation of additional health risk factor (air pollution, road trauma, NO2 and noise) will be discussed in the relevant code sections.

1.1.1 Difference between ITHIM and PMSLT

- **Time component** The *PMSLT* follows a population of interest over time. For example, as set up here, we simulate sex and age (5 years starting at 20) cohorts over time until they die or reach 100 years of age. This implies that we can include trends for diseases, time lags for change in exposure to risk factors and change in health and demographic changes (e.g. population growth). In addition, we can estimate yearly changes in the burden of diseases over the life course or for a specified number of years. The *ITHIM* approach is a snapshot of change in burden for one year.
- **Interaction between multiple diseases** The *PMSLT* accounts for the interaction between multiple diseases, with proportions of the population being able to be in more than one health state (Briggs,

Scarborough, and Smith 2016). This avoids overestimation of outcomes as a result of summing health outcomes attributable to each disease individually as done in *ITHIM*. It is important to note that the *PMSLT* assumes that diseases are independent of each other. That is to say, developing a disease is unrelated to a concurrent diagnoses of another disease).

- **Mortality rate** The *PMSLT* calculations for changes in life years (and health-adjusted life years) and mortality outcomes is based on observed mortality rates for the population of interest. In the *ITHIM* model, if burden of disease estimates from the Global Burden of Disease (GBD) study are used, then, the mortality component is based on the highest attained life expectancy observed in the world.
- **Impact of disability in increased life expectancy** In GBD studies, YLLs are not adjusted for disability; hence, their use in estimating intervention effects results in over-estimation, which the *PMSLT* approach avoids. Another way of seeing this is that estimated changes in morbidity using the *ITHIM* do not allow for how implicit increases in life expectancy impact on morbidity. While the changes in deaths and prevalence using the *PMSLT* are in some ways more accurate than those from the *ITHIM* approach it should be noted that that the average age of death and incident disease will change and thus the disease burden will be on average be shifted later in life (which is a realistic approach).

2 R development

The model is set up as a long script to perform the required mathematical calculations. Where possible, we wrote functions and loops to avoid repetition. We set up the model with data for Greater London. Figure 2 depicts the PMSLT model framework, which was followed in the code development.

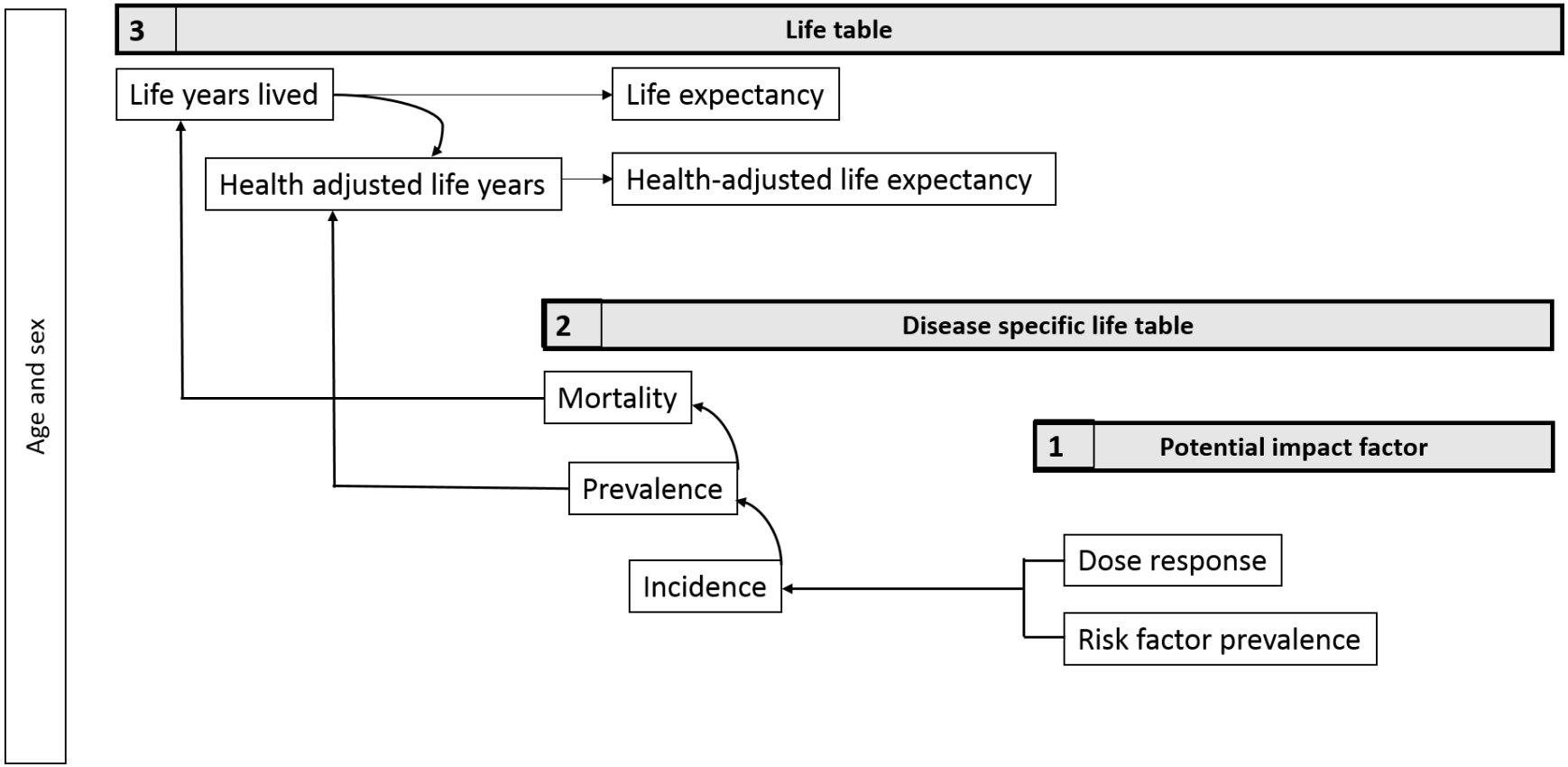

Figure 2: Proportional multi-state life-table simplified framework. *The simplied PMST shows the interaction between the life table, disease life table and potential impact fraction (PIF). The PIF calculations by age and sex group are the same as those generated for ITHIM. The PIF (or 1-PIF) modifies incidence of disease, which changes prevalence and mortality (disease specific life table). Changes in prevalence and mortality rates from the disease specific life tables feed into the life table by changing all-cause mortality, which in turn changes life years. Change in prevalence of diseases changes total years lived with disability, which in turn modifies health-adjusted life years*

In what follows, first, we specify input parameters. Second, we present the code with explaining notes. Third, we present examples of outcomes and lastly we comment on topics related to implementation. Here we only included the physical activity health pathway. In the comments section, we discuss the implementation of exposure to air pollution and road trauma. Note that in the presentation of input parameters, those needed to calculate PIFs are excluded, as these are common to the ITHIM, expect if trends are included (refer to comments section).

2.1 Inputs

We specify data requirements for the life table and disease life tables (Figure 2) and potential sources.

2.1.1 Life table

Inputs of the life table are: population numbers by sex (per 1-year or age grouping of interest), mortality rates or probability of all cause mortality by single age group and sex and total prevalent years lived with disability rate per single year by sex. Disease specific disability weights are presented as inputs here as these adjust the total years lived with disability, hence, the health-adjusted life years.

2.1.1.1 Population numbers

These data will be provided by the synthetic population or derived from other data. In the code presented here, we simulate 5-year age and sex cohorts. Data for population may be in five-year age groups or one-year. For the example for Greater London, we derive 5-year age groups from GBD IHME data, however, we also provide a script if five-year age groups are to be derived from one-year age groups data. I left potential data sources below as a reference.

Data source: (1) National census; (2) Worldwide population and mortality data: http://www.mortality.org/ (mostly high income countries; and (3) Calculate from the Global Burden of Disease by the Institute of Health Metrics and Evaluation (GBD IHME) data (rates and numbers available from (http://ghdx.healthdata.org/gbd-results-tool).

2.1.1.2 Mortality rates

Mortality rates are needed per single year and sex. These data are available from GBD IHME, however, in age groups (1-4, 5-9, etc). We provide an interpolation script to derive in between ages rates (cubic spline).

Note that we need data for population numbers and all cause mortality rates for: (1) PMSLT and (2) Dismod II collection (more in Dismod II section). Population data from the synthetic population is used for the PMSLT (if available). For Dismod II, population and mortality data should be from the same source (GBD IHME).

2.1.1.3 Total years lived with disability rates per single year and sex.

These data is available from the GBD (http://ghdx.healthdata.org/gbd-results-tool) per 5-year age groups. We can use interpolation to derive between ages rates (script provided).

2.1.1.4 Disability weights (quality of life weights)

Disability weights (DW) are derived from disease specific years lived with disability (YLD) and disease specific prevalence by age group (5 years) and sex. Data for YLDs prevalence is obtained from the online GBD IHME data tool (http://ghdx.healthdata.org/gbd-results-tool). An age and sex specific-correction was introduced to counteract the effects of accumulating comorbid illnesses in the older age groups (Equation 1).

$$(YLDd/Pd)/(1 - YLDt) = DWadjustedfortotalYLDs \quad (1)$$

Where YLDd is the YLD mean number per age and sex for a given disease, Pd is the prevalence (as reported in GBD) for a given disease by age and sex and YLDt is total YLD rate per age and sex.

2.1.2 Disease life tables

2.1.2.1 Incidence and case fatality

For each of the modeled diseases the PMSLT needs incidence and case fatality rates per sex and one-year intervals. Data from the GBD IHME studies with Dismod II (free at https://www.epigear.com/index_files/dismod_ii.html) is used to derive internally consistent data and generate missing data. For example, the GBD studies provide data for incidence, prevalence and disease mortality, however, not case fatality. Other national level sources may also be explored/used, and compare with estimates produce from GBD data and Dismod II.

Dismod II inputs are: (1) population numbers and mortality rates and (2) disease specific inputs.

Population and mortality

Within Dismod II, each setting (e.g. country) has a collection that consists of population numbers (preferably the same as used in GBD IHME studies, due to the mortality envelop) and all- cause mortality rates (numbers and calculate rates). The GBD provides 5-year age groups that are acceptable input parameters for Dismod II.

Disease inputs by age group and sex

Each setting collection has a given number of diseases. Dismod II works with at least three of: case fatality, prevalence, incidence, mortality (disease), case fatality, remission, duration and the relative risk for mortality. So far, we have been assuming that remission is zero for chronic diseases, that is to say, when people become diseased, they do not recover. Special care should be taken with this assumption, as the GBD data assumes remission for some diseases, for example cancers, where after 10 years cases recover, except for long term sequelae. Since GBD now provides prevalence, incidence and mortality, it may be best to use all three as Dismod II input parameters to compare the effect of the remission assumption by the GBD for some diseases.

2.2 Code

Following the structure of Figure 2, we developed functions to perform sex and age cohorts calculations for the life table, disease life tables and potential impact fractions: run_life_table, run_disease and and run_pif. We also generated two functions for outputs: plot_outputs and gen_aggregate. The function plot_outputs

Table 1: PMSLT inputs

Input	Source	Comments
Life table	Synthetic population per sex and age group	Age grouping in life table to match synthetic population
Life table	Synthetic population per sex and one-year age group	If one year age group is not avabilable it can be derive using interpolation from age groups data
Life table	Global Burden of Disease (GBD) study per one-year age group and sex	GBD data is in five-year age groups, interpolation to derive one-year age groups
Disease life table	GBD data for prevalence, incidence and mortality and DISMOD II	Two step process. First obtain disease and population data from GBD. Second, use Dismod II to derive internally consistent estimates for incidence and case fatality (PMSLT disease life table iputs)
Disease life table	Derive from disease prevalence and years lived with disability from GBD	Adjustments for comorbidities in later years of life to be applied

creates age-group and sex linear plots for specified outcomes (e.g. health-adjusted life years, incidence of diabetes) and gen_aggregate adds up each cohort results. Functions were then used in a code script. In what follows, we explain each step in the development of the script. Here we also include code chunks, however, we also kept them separately in the MSLT folder, in the code file.

In what follows, we start explain the script step by step.

2.2.1 Set up

We start by cleaning the global environment (1) to keep track of our works and ensure that the code is generating our desired outcomes. Then, we set up an option to avoid the use of scientific notation (2) and lastly we load the functions (3). The code chunks are shown in the rmarkdown output.

```
knitr::read_chunk('mslt_code.r')
# source("code/functions.R")
```

1) Clean Global Environment

```
rm (list = ls())
```

2) Avoid scientific notation

```
options(scipen=999)
```

3) Load functions

```
source("code/functions.R")
```

2.2.2 Inputs

Table 1 describes data needs for the PMSLT, here we expand on the data needs and mechanisms (Figure 3) to use the PMSLT approach in ITHIMR (Figure 1).

Initial case studies for the ITHIMR are: London, Sao Pablo, Delhi, Accra, Los Angeles and Edinburgh. Here, we will start with **Greater London** given the availability of disease epidemiology data from the GBD IHME study. For the rest of the case study cities data is available at the country level, hence, a scaling method is needed to reflect the local burden of disease.

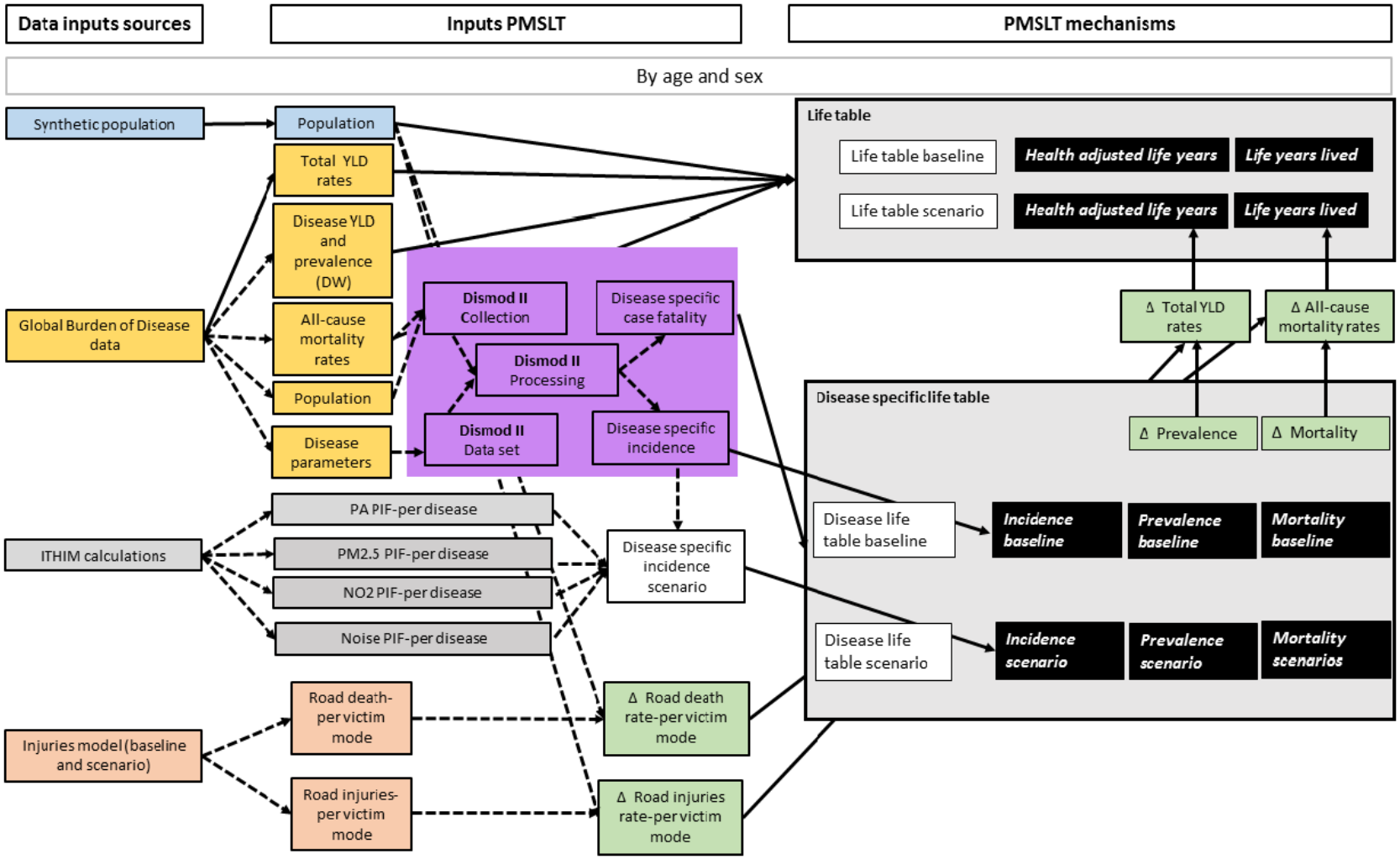

Figure 3: Proportional multi-state life-table model. *Three sections are presented in Figure 3:* ***Data input sources***, ***Inputs PMSLT*** *and* ***PMSLT mechanisms***. *The color coding from Data inputs sources to Inputs PMSLT link sources with inputs for the PMSLT. Solid arrows represent final inputs and dashed-arrows represent intermediate inputs that need further processing. Purple coding means a process and green coding represent change in mortality and disability prevalence rates to modify the life table parameters. Black color coding with white color font represent final model outcomes. For both, the life table and disease life table, two sets of each are simulated, one for the baseline and the other for the scenario.*

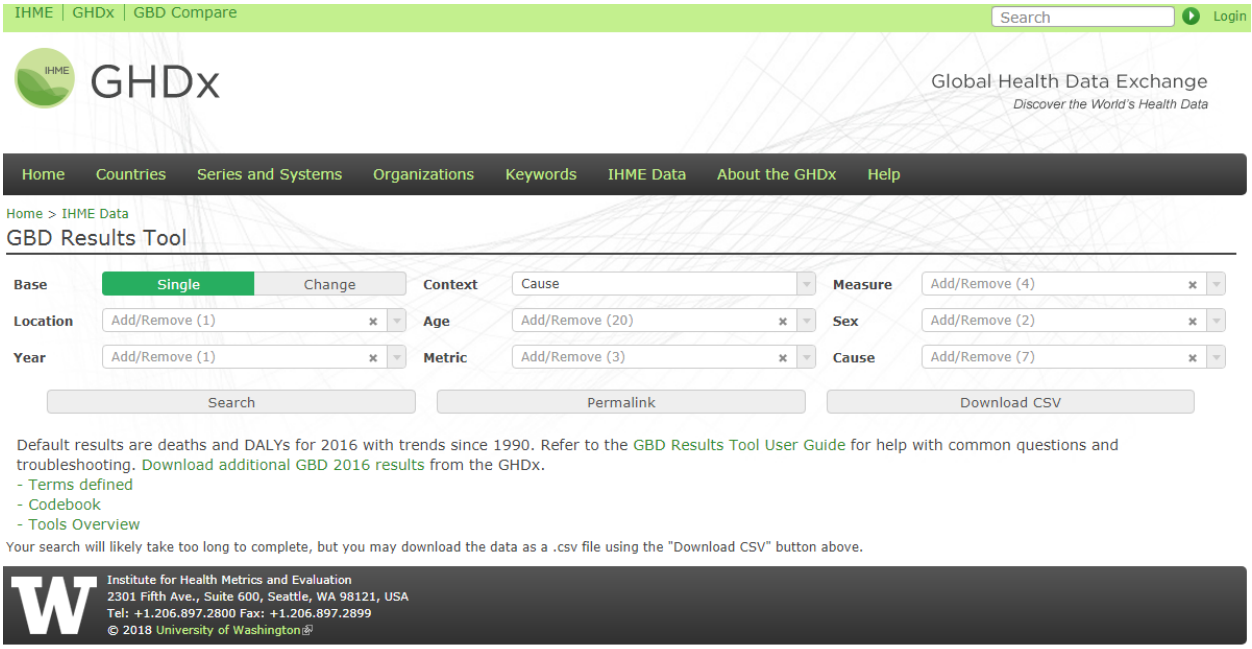

Figure 4: Global Burden of Disease data results tool.

2.2.2.1 Global Burden of Disease data

First, we explain how to obtain the data, second additional processing to derive data not reported (population) and one-year age groups (original data is in five-year age groups) and last procedure to use Dismod II. Data from the *Global Burden of Disease data* in Figure 3 can be download from here: http://ghdx.healthdata.org/gbd-results-tool. Figure 4 is a screenshot of the GHDx.

Table 2 specifies the selections to do for each of the tabs in Figure 2.

Once the selections described in Table 2 are made, the option *Download CVS** in the GHDx website is selected. A prompt comes up asking for an email address. The data is sent to the designated email address (within minutes) in ZIP format, unzip and use the code below to read the data (4). Here, we selected data for Greater London and England. The aim is to compare and derive scaling factors as for most cities the data is not available from the GBD and country level data may be used and scaled to the city level. Note that the data input requirement for the PMSLT, except population numbers, is in rates. Therefore the scaling is to better reflect the burden of an area, this is a different issue than working with numbers (e.g. total mortality numbers, total YLDs numbers) as in the ITHIM approach.

4) Read GBD data

```
GBDdata <- read.csv("data/legacy/UK/gbd2016.csv", stringsAsFactors = F)
```

The following codes serves to sort out the GBD data to the inputs required for the life table, disease life table and Dismod II

These data should be used to generate the general life table and disease life tables (Figure 3).

5) Change all upper cases to lower case and delete () from variables.

Table 2: Global burden of disease data

Tab	Selection
Base	Single
Location	Case study city
Year	Latest available
Context	Cause
Age	Under 5, 5 to 9, 10 to 14, 15 to 19, 20 to 24, 25 to 29, 30 to 34, 35 to 39, 40 to 49, 50 to 54, 55 to 59, 60 to 64, 65 to 69, 70 to 74, 75 to 79, 80 to 84, 89 to 89, 90 to 95, 95 plus
Metric	Number, Rate
Measure	Deaths, YLDs, Prevalence, Incidence
Sex	Male, Female
Cause	Total All causes, ischemic heart disease, etc

```
GBDdata <- mutate_all(GBDdata, funs(tolower))
GBDdata$measure[GBDdata$measure=="ylds (years lived with disability)"] <- "ylds"
```

6) Create age categories index in GBDdata (mid age, to match cohort running), total of 20 age groups. These are the age cohorts to simulate.

```
# 22, 27, 32, 37, 42, 47, 52, 57, 62, 67, 72, 77, 82, 87, 92, 97
GBDdata$age_cat [GBDdata$age =="under 5"] <- 2
GBDdata$age_cat [GBDdata$age =="5 to 9"] <- 7
GBDdata$age_cat [GBDdata$age =="10 to 14"] <- 12
GBDdata$age_cat [GBDdata$age =="15 to 19"] <- 17
GBDdata$age_cat [GBDdata$age =="20 to 24"] <- 22
GBDdata$age_cat [GBDdata$age =="25 to 29"] <- 27
GBDdata$age_cat [GBDdata$age =="30 to 34"] <- 32
GBDdata$age_cat [GBDdata$age =="35 to 39"] <- 37
GBDdata$age_cat [GBDdata$age =="40 to 44"] <- 42
GBDdata$age_cat [GBDdata$age =="45 to 49"] <- 47
GBDdata$age_cat [GBDdata$age =="50 to 54"] <- 52
GBDdata$age_cat [GBDdata$age =="55 to 59"] <- 57
GBDdata$age_cat [GBDdata$age =="60 to 64"] <- 62
GBDdata$age_cat [GBDdata$age =="65 to 69"] <- 67
GBDdata$age_cat [GBDdata$age =="70 to 74"] <- 72
GBDdata$age_cat [GBDdata$age =="75 to 79"] <- 77
GBDdata$age_cat [GBDdata$age =="80 to 84"] <- 82
GBDdata$age_cat [GBDdata$age =="85 to 89"] <- 87
GBDdata$age_cat [GBDdata$age =="90 to 94"] <- 92
GBDdata$age_cat [GBDdata$age =="95 plus"] <- 97
```

7) Create age and sex categories to obtain population numbers. Population numbers from GBD are used in Dismod II. For the Life table (Figure 3), the numbers may be from the synthetic population. For

now, the Life table is set up with population numbers derived from the GBD data.

```
GBDdata$sex_age_cat <- paste(GBDdata$sex,GBDdata$age_cat, sep = "_"  )
```

8) Convert string variables to numeric to do calculations.

```
GBDdata$val <- as.numeric(as.character(GBDdata$val))
```

9) Generate population numbers for Greater London in a new data frame ("GBD_population"). Note that there is data for England as well, which is used in a separate rmarkdown document (GBDCompare). Population numbers are derived from rates per 100,000 and total numbers of cases.

```
GBD_population <- filter(GBDdata, measure == "deaths", cause == "all causes",
                         metric == "rate" | metric == "number" ) %>% select(metric, age_cat, sex, val,
                                                                            sex_age_cat, location)
```

10) Generate population numbers from given number of cases and rates per 100,000 people.

```
for (i in 1:nrow(GBD_population)) {
  if (GBD_population$metric[i] == "number") {
    GBD_population$five_year_population[i] <- GBD_population$val[i] * 100000/
      GBD_population$val[i + 2]}
  else {GBD_population$five_year_population[i] <- NA}
}
```

11) Remove rows with zero

```
GBD_population <- GBD_population[!is.na(GBD_population$five_year_population),]
```

12) Keep relevant variables

```
GBD_population <- filter(GBD_population) %>%
  select(sex_age_cat, sex, age_cat, five_year_population, location)
```

13) Create data frames for Greater London to be later used for: a) interpolation of rates and b) PMLT cohorts.

```
GBD_population_GL <-  filter(GBD_population, location == "greater london") %>%
  select(sex_age_cat, age_cat, sex, five_year_population, location, sex_age_cat)
```

14) Check population total numbers

```
GreaterLondon <- sum(GBD_population_GL$five_year_population)
```

15) Generate data frames for Greater London with per person rates (per 100,000 in original data).

```
GBDGL <- filter(GBDdata, location == "greater london" & metric == "rate") %>%
  select(measure, location, sex, age, metric, cause, val, age_cat, sex_age_cat)
GBDGL$one_rate <- GBDGL$val/100000
```

16) For the life table, we need to mortality and total yld rates in one year age intervals. The original data is in five years. Thus, the following code is used to interpolate a single-year age distribution form a

five-yearly distribution.

```
## Loop to generate interpolated rates for all cause mortality and ylds for males and females
## UPDATE WITH YLDS RATES ALL CAUSES ADJUSTED FOR ALL OTHER MODELLED DISEASES.

i_sex <- c("male", "female")
i_measure <- c("deaths", "ylds") #" (years lived with disability)")

index <- 1

for(sex_index in i_sex) {
  for (measure_index in i_measure) {

    data <- filter(GBDGL, measure == measure_index, sex == sex_index,
                   cause == "all causes") %>% select(measure, location, sex, age, metric,
                                                     cause, val, age_cat, one_rate)
    assign(paste(sex_index, measure_index, "interpolated_data", sep = "_"), data)
    x=data$age_cat
    y=log(data$one_rate)

    interpolation_func <- splinefun(x, y, method = "natural", ties = mean)

    interpolated <- as.data.frame(interpolation_func(seq(0, 100, 1)))
    age <- seq(0,100,by=1)
    interpolated <- cbind(interpolated, age)
    interpolated[,1] <- exp(interpolated[,1])
    colnames(interpolated)[1] <- paste(measure_index)
    ## Add column with sex to create age_sex category to then merge with input_life table
    interpolated$sex <- paste(sex_index)
    interpolated$sex_age_cat <- paste(interpolated$sex, interpolated$age, sep = "_")
    ## Change name of column death to mx and ylds to pyld_rate to then merge
    ## with input_life table

    if (colnames(interpolated)[1] == "deaths")
      colnames(interpolated)[1] <- paste("mx")
    else
      colnames(interpolated)[1] <- paste("pyld_rate")

    # Name data frame
    assign(paste(sex_index, measure_index, "interpolated", sep = "_"), interpolated)

  }
}
```

```
## Do graph with another layer for original rates in age groups for comparison purposes.

p_interpolation_list <- list()

interpolation_index <- 1

for(sex_index in i_sex) {
  for (measure_index in i_measure) {

    p_interpolation_index <- ggplot(data = interpolated,mapping = aes(age, interpolated[,1])) +
      geom_line(aes(color = "Interpolated")) +
      geom_point(
        data = data,
        mapping = aes(age_cat, one_rate, color = "Original")) +
      labs(colour="",x="Age",y="Rates", sep = " ") +
      labs (title = paste("Rates", sex_index, "all cause",
                          measure_index, sep = " "), size=14) +
      theme_classic() +
      theme (plot.title = element_text(hjust = 0.5))

    p_interpolation_list[[interpolation_index]] <- p_interpolation_index
    interpolation_index <- interpolation_index + 1

  }
}

## Save plots to jpeg. Makes a separate file for each plot.

interpolation_index <- 1
for(sex_index in i_sex) {
  for (measure_index in i_measure) {
    file_name = paste("output/graphs1/", "Interpolation rates", sex_index, measure_index, ".jpeg", sep=
    jpeg(file_name)
    print(p_interpolation_list[[interpolation_index]])
    interpolation_index <- interpolation_index + 1
    dev.off()
  }
}


p_interpolated <- do.call(marrangeGrob, list(grobs=p_interpolation_list, nrow = 2, ncol = 2))
p_interpolated
```

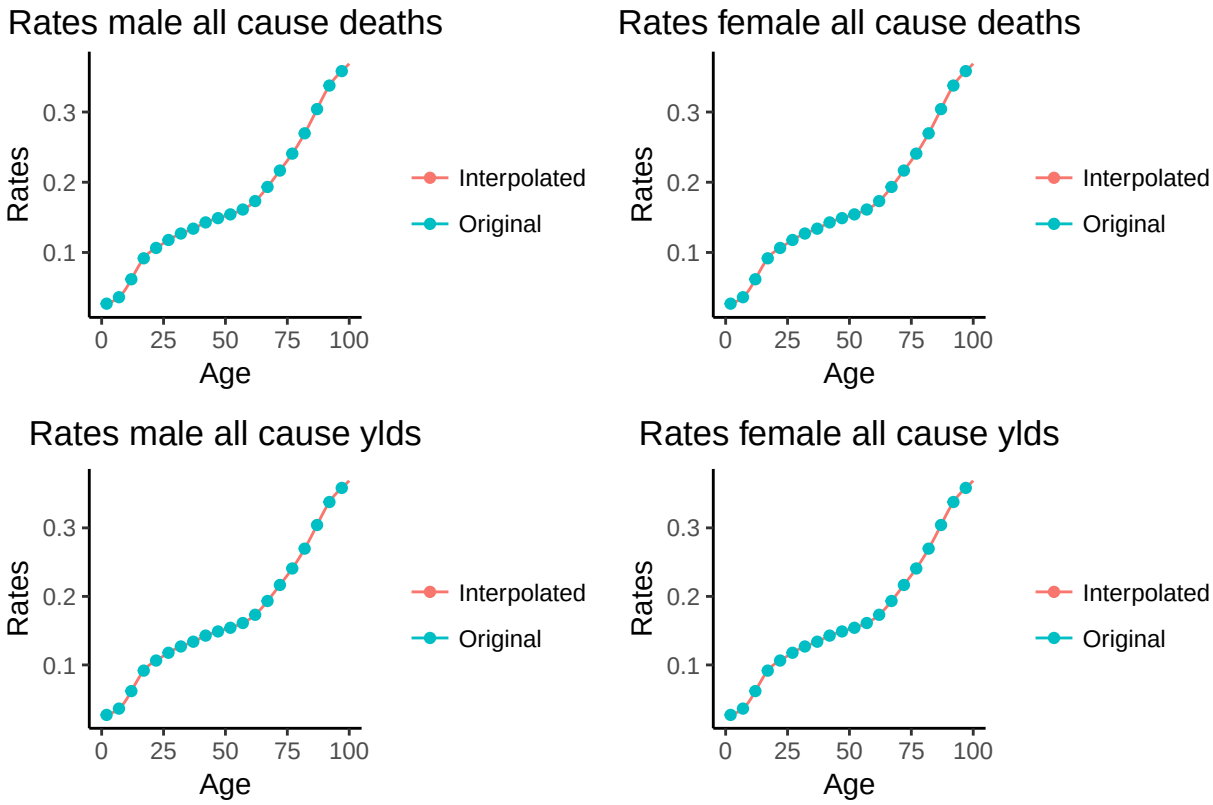

Figure 5: FALSE

17) The following code brings together all the required data to create the inputs life table (Figure 3).

```
## Create empty data frame

input_life_table <- as.data.frame(matrix(0, ncol=2, nrow = 202))

## Give variables names (mx=total mortality rate,
## yldsx = total years lived with disability rates )

names(input_life_table) <- c("age", "sex")

## Populate life table: Age, 0 to 100

input_life_table[1:101, 1] <- c(0:100)
input_life_table[102:202, 1] <- c(0:100)

## Populate life table: sex, male, female

input_life_table[1:101, 2] <- "male"
```

```r
input_life_table[102:202, 2] <- "female"

## Create variable age_sex to match with population data

input_life_table$sex_age_cat <- paste(input_life_table$sex,input_life_table$age,
                                      sep = "_"  )
## Populate life table: five_year_pop.

input_life_table <- merge(input_life_table, select(GBD_population_GL,
                                                   c(sex_age_cat, five_year_population)), by = 'sex_age_

## Populate life table: mortality rates (from interpolated rates)
## Males mortality rates. NEED MATCHING NAMES

input_life_table <- merge(input_life_table, select(male_deaths_interpolated,
                                                   c(sex_age_cat, mx)), by = 'sex_age_cat', all.x = TRU
input_life_table <- merge(input_life_table, select(female_deaths_interpolated,
                                                   c(sex_age_cat, mx)), by = 'sex_age_cat', all.x = TRU
input_life_table <- merge(input_life_table, select(male_ylds_interpolated,
                                                   c(sex_age_cat, pyld_rate)) , by = 'sex_age_cat', all
input_life_table <- merge(input_life_table, select(female_ylds_interpolated,
                                                   c(sex_age_cat, pyld_rate)), by = 'sex_age_cat', all.

## DISCUSS WITH ALI TO IMPROVE

input_life_table$mx <-  ifelse(input_life_table$sex == "male",
                               input_life_table$mx.x, input_life_table$mx.y)

input_life_table$pyld_rate <-  ifelse(input_life_table$sex == "male",
                                      input_life_table$pyld_rate.x, input_life_table$pyld_rate.y)
## Drop redundant variables
input_life_table <- subset(input_life_table, select = -c(mx.y, mx.x, pyld_rate.x, pyld_rate.y))

## Cross check population numbers

sum(input_life_table$five_year_population, na.rm = TRUE)
```

```
## [1] 8778396
```

```r
## Sort data by sex and age to use in life table function

input_life_table<-input_life_table[order(input_life_table$sex, input_life_table$age),]
```

18) Generate **baseline life tables** from input_data_frame using run_life_table function. Life tables are

generated for each age (5-years) and sex cohort. It is assumed that current observed rates of mortality and disability are going to be observed in the future.

```
i_age_cohort <- c(22, 27, 32, 37, 42, 47, 52, 57, 62, 67, 72, 77, 82, 87, 92, 97)

general_life_table_list_bl <- list()

index <- 1

for (age in i_age_cohort){
  for (sex in i_sex){
    # cat("age ", age, " and sex ", sex, "\n") #Uncomment to see index
    general_life_table_list_bl[[index]] <- run_life_table(in_idata = input_life_table,
                                                          in_sex = sex, in_mid_age = age)
    index <- index + 1
  }
}

## Uncommnet to check life table list
# View(general_life_table_list_bl[[32]])
```

19) Generate **baseline disease life tables** using run_disease function. Inputs of the disease life tables are: incidence, case fatality (Figure 3, purple shaded area) and disability weights. Internally consistent estimates of incidence and case fatality are derived from GBD data (by five-year age groups and sex) and Dismod II. Disability weights are derived from GBD disease specific YLDs and prevalence (by five-year age groups and sex) adjusted for all-cause YLDs (see section 2.1.1.4).
Note that Dismod II is an external software. The data generation for Dismod II is now done externally and the end product, for the example of Greater London is idata. For now, we plan to get the data processing for all data for the MSLT and Dismod using code (space blank left below). We now have all the data needs in the main data folder in data.

```
## Use externaly generated inputs for for disease life table. The previous code will serve to to some o

idata <- read.csv("data/legacy/UK/idata.csv", stringsAsFactors = F)
## Use run_disease

i_disease <- c("ihd", "istroke", "diabetes", "colon_cancer", "breast_cancer")


disease_life_table_list_bl <- list()
index <- 1

for (age in i_age_cohort){
  for (sex in i_sex){
    for (disease in i_disease) {
```

```
      # Exclude breast_cancer for Males
      if (sex == "male" && disease == "breast_cancer"){
        # cat("\n") #Uncomment to see list
      }
      else {
        # cat("age ", age, " sex ", sex, "and disease", disease, "\n") #Uncomment to see list
        disease_life_table_list_bl[[index]] <- run_disease(in_idata = idata, in_sex = sex, in_mid_age =
        index <- index + 1
      }
    }
  }
}
## Uncommnet to check disease life table list
# View(disease_life_table_list_bl[[8]])
```

20) Generate mock change in incidence of disease to generate scenario life tables. In the final version, this will come from the calculated PIFs by disease. PIFs are applied here to incidence, however, may also be applied to case fatality, depending on the RRs. Also, we can include time delays from change in exposure to change in health outcomes via the PIF.

```
## Create value to use as factor changing incidence rates.

incidence_change <- 0.95

## Generate scenario incidence (for each disease)

incidence_sc <- list()
index <- 1

for (age in i_age_cohort){
  for (sex in i_sex){
    for (disease in i_disease) {

      # Exclude breast_cancer for Males
      if (sex == "male" && disease == "breast_cancer"){
        # cat("\n") # Uncomment to see list
      }
      else {

        incidence_sc[[index]] <- disease_life_table_list_bl[[index]]$incidence_disease *
          incidence_change
        index <- index + 1

      }
```

```
    }
  }
}

## Uncommnet to check scenario incidence
# View(incidence_sc[[1]])
```

21) Use scenario incidence to calculate scenario disease life tables.

```
disease_life_table_list_sc <- list()
index <- 1
for (age in i_age_cohort){
  for (sex in i_sex){
    for (disease in i_disease) {
      # Exclude breast_cancer for Males
      if (sex == "male" && disease == "breast_cancer"){
        # cat("\n")
      }
      else {
        # cat("age ", age, " sex ", sex, "and disease", disease, "\n")
        # modify idata's incidence for the said scenario
        td1 <- idata
        td1[td1$age >= age & td1$sex == sex,][[paste("incidence", disease, sep = "_")]] <- incidence_sc

        # Instead of idata, feed td to run scenarios
        disease_life_table_list_sc[[index]] <- run_disease(in_idata = td1, in_sex = sex,
                                                           in_mid_age = age, in_disease = disease)
        disease_life_table_list_sc[[index]]$diff_inc_disease <-
          disease_life_table_list_sc[[index]]$incidence_disease -   disease_life_table_list_bl[[index]]$
        disease_life_table_list_sc[[index]]$diff_prev_disease <-
          disease_life_table_list_sc[[index]]$px  - disease_life_table_list_bl[[index]]$px
        disease_life_table_list_sc[[index]]$diff_mort_disease <-
          disease_life_table_list_sc[[index]]$mx - disease_life_table_list_bl[[index]]$mx
        disease_life_table_list_sc[[index]]$diff_pylds_disease <-
          (disease_life_table_list_sc[[index]]$px - disease_life_table_list_bl[[index]]$px) * disease_li
        index <- index + 1
      }
    }
  }
}
## Uncommnet to check scenario life tables
# View(disease_life_table_list_sc[[1]])
```

22) Calculate life tables scenario.

```
## Generate total change in mortality rate

## Sum mortality rate change scenarios (mx_sc_total)

mx_sc_total <- list()
l_index <- 1
index <- 1
for (age in i_age_cohort){
  for (sex in i_sex){
    mortality_sum <- NULL
    create_new <- T

    for (disease in i_disease) {
      if (sex == "male" && disease == "breast_cancer"){
        # cat("\n")
      }else{

        if (create_new){
          mortality_sum <- select(disease_life_table_list_sc[[index]],
                                  c('age', 'sex'))
          mortality_sum$total <- 0
          create_new <- F
          mortality_sum$total <- mortality_sum$total +
            (disease_life_table_list_sc[[index]]$diff_mort_disease)
        }else{
          mortality_sum$total <- mortality_sum$total +
            (disease_life_table_list_sc[[index]]$diff_mort_disease)
        }

        # cat(age, " - ", sex," - ",  disease," - ",  index, " - ", l_index,  "\n")
        index <- index + 1
      }
    }
    mx_sc_total[[l_index]] <- mortality_sum

    l_index <- l_index + 1
  }
}

## Uncommnet to check sceanrio mortality and changes
# View(mx_sc_total[[3]])

## Generate total change in prevalent yld rates
```

```r
## Total ylds rate= sum (change prevalence disease*dw)

pylds_sc_total <- list()
l_index <- 1
index <- 1
for (age in i_age_cohort){
  for (sex in i_sex){
    pylds_sum <- NULL
    create_new <- T

    for (disease in i_disease) {
      if (sex == "male" && disease == "breast_cancer"){
        # cat("\n")
      }else{

        if (create_new){
          pylds_sum <- select(disease_life_table_list_sc[[index]], c('age', 'sex'))
          pylds_sum$total <- 0
          create_new <- F
          pylds_sum$total <- pylds_sum$total +
            (disease_life_table_list_sc[[index]]$diff_pylds_disease)
        }else{
          pylds_sum$total <- pylds_sum$total +
            (disease_life_table_list_sc[[index]]$diff_pylds_disease)
        }

        # cat(age, " - ", sex," - ",  disease," - ",  index, " - ", l_index,  "\n")
        index <- index + 1
      }

    }
    pylds_sc_total[[l_index]] <- pylds_sum
    l_index <- l_index + 1
  }
}

## Uncommnet to check scenario pyld change
# View(pylds_sc_total[[2]])

## Calculate general life tables with modified mortality and pylds total
## Original mortality rate is modified by the mx_sc_total (total change in mortality from diseases)
## Original pyld rate is modified by the change in each disease pylds
```

```r
general_life_table_list_sc <- list()
index <- 1


for (age in i_age_cohort){
  for (sex in i_sex){

    cat("age ", age, " and sex ", sex, "\n")
    # modify idata's mortality and pyld total for the said scenario
    td2 <- input_life_table
    # td2 <- subset(td2, select = -c(mx, pyld_rate))
    td2[td2$age >= age & td2$sex == sex,][[paste("mx")]] <- general_life_table_list_bl[[index]]$mx + mx
    td2[td2$age >= age & td2$sex == sex,][[paste("pyld_rate")]] <- general_life_table_list_bl[[index]]$

    # Instead of idata, feed td to run scenarios
    general_life_table_list_sc[[index]] <- run_life_table(in_idata = td2, in_sex = sex, in_mid_age = ag
    #

    index <- index + 1
  }
}
```

```
## age  22  and sex  male
## age  22  and sex  female
## age  27  and sex  male
## age  27  and sex  female
## age  32  and sex  male
## age  32  and sex  female
## age  37  and sex  male
## age  37  and sex  female
## age  42  and sex  male
## age  42  and sex  female
## age  47  and sex  male
## age  47  and sex  female
## age  52  and sex  male
## age  52  and sex  female
## age  57  and sex  male
## age  57  and sex  female
## age  62  and sex  male
## age  62  and sex  female
## age  67  and sex  male
## age  67  and sex  female
```

```
## age  72  and sex  male
## age  72  and sex  female
## age  77  and sex  male
## age  77  and sex  female
## age  82  and sex  male
## age  82  and sex  female
## age  87  and sex  male
## age  87  and sex  female
## age  92  and sex  male
## age  92  and sex  female
## age  97  and sex  male
## age  97  and sex  female
```

```
## Uncommnet to check scenario life tables
# View(general_life_table_list_sc[[32]])
# View(general_life_table_list_bl[[1]])

## Check difference life table baseline and scenario
general_life_table_list_bl[[1]]$Lx - general_life_table_list_sc[[1]]$Lx
```

```
##  [1]    -0.001104913    -0.005962362    -0.016622719    -0.034760004
##  [5]    -0.062078574    -0.100895857     0.083088827     0.539253059
##  [9]     0.839619895     0.925748932     0.977418006     0.983653806
## [13]     0.931499487     0.811214541     0.610232388     0.283640733
## [17]    -0.228501673    -0.964645937    -1.953609671    -3.232122140
## [21]    -4.846187985    -6.848005185    -9.304768090   -12.319771308
## [25]   -15.978455105   -20.358743412   -25.585010520   -31.725278121
## [29]   -38.862890669   -47.086995598   -56.455213916   -67.056150209
## [33]   -78.949307084   -92.181211507  -106.861688245  -123.115224182
## [37]  -141.002933288  -160.629078539  -182.132458584  -205.594983017
## [41]  -231.131408268  -258.882650131  -288.889299253  -321.088557837
## [45]  -355.443234499  -392.040832169  -430.878386097  -471.986181242
## [49]  -515.379002628  -560.924351198  -608.502571008  -657.921568353
## [53]  -708.951062802  -761.172108152  -813.904366388  -866.172169674
## [57]  -916.923861322  -965.376056454 -1010.509415458 -1051.145336369
## [61] -1085.914560197 -1113.167205451 -1131.221187177 -1139.332320444
## [65] -1137.577838258 -1125.506468909 -1102.834165304 -1069.489408401
## [69] -1025.314747023  -971.646143801  -910.104355705  -840.930121897
## [73]  -764.213389167  -680.545928524  -591.311294031  -498.451988916
## [77]  -405.269002831  -301.779728986  -571.530089454
```

```
general_life_table_list_bl[[1]]$Lwx - general_life_table_list_sc[[1]]$Lwx
```

```
##  [1]    -0.2220033    -0.7120024    -1.2503428    -2.0292190    -2.8221633
##  [6]    -3.7703351     1.5247129    13.0984652    20.8509341    19.4155442
```

```
## [11]    17.7842581    15.8389993    13.6219269    12.5803878     9.9595939
## [16]     6.8220922     3.0655452    -1.1397565    -5.8087889   -11.1246292
## [21]   -17.0932941   -23.6904242   -31.1679920   -36.6563902   -45.5698718
## [26]   -55.5149300   -66.6004981   -78.7999645   -92.9288638  -107.7392474
## [31]  -123.8541533  -141.4060835  -160.4278521  -181.0027578  -203.1410081
## [36]  -226.7958554  -251.9808062  -278.8052728  -308.6913410  -338.8973988
## [41]  -370.7385191  -404.2409769  -439.2258179  -464.4336516  -501.6349447
## [46]  -540.0207567  -579.4161739  -619.6748456  -673.1755309  -715.2885316
## [51]  -757.7323445  -800.2036299  -842.3510348  -910.8421573  -950.7869210
## [56]  -988.1394672 -1021.9472808 -1051.2581180 -1089.7404019 -1107.1186438
## [61] -1116.7133980 -1117.3177380 -1108.0502139 -1121.9790078 -1091.0886078
## [66] -1051.0140180 -1002.4910789  -946.5314825  -910.7811344  -839.7382719
## [71]  -765.7615923  -689.5698643  -611.4916513  -552.5862962  -467.9645274
## [76]  -385.1109562  -306.2007535  -225.0756266  -369.7788610
```

23) Generate list of outputs by age and sex.

```
## In the following list "output_life_table", 32 data frames are nested per age and sex cohort

output_burden <- list()
l_index <- 1
index <- 1
for (age in i_age_cohort){
  for (sex in i_sex){

    # Males do not have breast cancer, that is why we need the if/else.
    # We create a TRUE/FALSE variable for the loop to move into the next disease

    create_new <- T
    for (disease in i_disease) {
      if (sex == "male" && disease == "breast_cancer"){
        # cat("\n")
      }else{

        if (create_new){
          output_burden_sc <- select(disease_life_table_list_sc[[index]],
                                     c('age', 'sex', 'incidence_disease', 'mx', 'px'))
          names(output_burden_sc)[names(output_burden_sc) == 'incidence_disease'] <-
            paste('incidence_disease', disease, "sc", sep = "_")
          names(output_burden_sc)[names(output_burden_sc) == 'mx'] <-
            paste('mx', disease, "sc", sep = "_")
          names(output_burden_sc)[names(output_burden_sc) == 'px'] <-
            paste('px', disease, "sc", sep = "_")
          output_burden_bl <- select(disease_life_table_list_bl[[index]],
```

```
                                c('incidence_disease', 'mx', 'px'))
  names(output_burden_bl)[names(output_burden_bl) == 'incidence_disease'] <-
    paste('incidence_disease', disease, "bl", sep = "_")
  names(output_burden_bl)[names(output_burden_bl) == 'mx'] <-
    paste('mx', disease, "bl", sep = "_")
  names(output_burden_bl)[names(output_burden_bl) == 'px'] <-
    paste('px', disease, "bl", sep = "_")

  ## New list to add calculations for changes in burden of disease (incidence and mortality numb

  output_burden_change <- list()

  output_burden_change$inc_num_bl <- disease_life_table_list_bl[[index]]$incidence_disease *
    (1 - disease_life_table_list_bl[[index]]$px) * general_life_table_list_bl[[l_index]]$Lx
  output_burden_change$inc_num_sc <- disease_life_table_list_sc[[index]]$incidence_disease *
    (1 - disease_life_table_list_sc[[index]]$px) * general_life_table_list_sc[[l_index]]$Lx
  output_burden_change$inc_num_diff <- (disease_life_table_list_sc[[index]]$incidence_disease *
                                          (1 - disease_life_table_list_sc[[index]]$px) * general



  output_burden_change$mx_num_bl <- disease_life_table_list_bl[[index]]$mx * general_life_table_
  output_burden_change$mx_num_sc <- disease_life_table_list_sc[[index]]$mx * general_life_table_
  output_burden_change$mx_num_diff <- (disease_life_table_list_sc[[index]]$mx * general_life_tab

  names(output_burden_change)[names(output_burden_change) == 'inc_num_bl'] <-
    paste('inc_num_bl', disease, sep = "_")
  names(output_burden_change)[names(output_burden_change) == 'inc_num_sc'] <-
    paste('inc_num_sc', disease, sep = "_")
  names(output_burden_change)[names(output_burden_change) == 'inc_num_diff'] <-
    paste('inc_num_diff', disease, sep = "_")
  names(output_burden_change)[names(output_burden_change) == 'mx_num_bl'] <-
    paste('mx_num_bl', disease, sep = "_")
  names(output_burden_change)[names(output_burden_change) == 'mx_num_sc'] <-
    paste('mx_num_sc', disease, sep = "_")
  names(output_burden_change)[names(output_burden_change) == 'mx_num_diff'] <-
    paste('mx_num_diff', disease, sep = "_")

  ## Bind all lists

  output_burden_sc <- cbind(output_burden_sc, output_burden_bl)
  output_burden_sc <- cbind(output_burden_sc, output_burden_change)

  create_new <- F
```

```
    ## Here the calculations above are repeated, here is where the F is telling to move into the n

  }else{

    td3 <- select(disease_life_table_list_sc[[index]],
                  c('incidence_disease', 'mx', 'px'))
    names(td3)[names(td3) == 'incidence_disease'] <-
      paste('incidence_disease', disease, "sc", sep = "_")
    names(td3)[names(td3) == 'mx'] <-
      paste('mx', disease, "sc", sep = "_")
    names(td3)[names(td3) == 'px'] <-
      paste('px', disease, "sc", sep = "_")

    td4 <- select(disease_life_table_list_bl[[index]],
                  c('incidence_disease', 'mx', 'px'))
    names(td4)[names(td4) == 'incidence_disease'] <-
      paste('incidence_disease', disease, "bl", sep = "_")
    names(td4)[names(td4) == 'mx'] <-
      paste('mx', disease, "bl", sep = "_")
    names(td4)[names(td4) == 'px'] <-
      paste('px', disease, "bl", sep = "_")

    output_burden_change2 <- list()

    output_burden_change2$inc_num_bl <- disease_life_table_list_bl[[index]]$incidence_disease * (
    output_burden_change2$inc_num_sc <- disease_life_table_list_sc[[index]]$incidence_disease * (
    output_burden_change2$inc_num_diff <- (disease_life_table_list_sc[[index]]$incidence_disease

    output_burden_change2$mx_num_bl <- disease_life_table_list_bl[[index]]$mx * general_life_table
    output_burden_change2$mx_num_sc <- disease_life_table_list_sc[[index]]$mx * general_life_table
    output_burden_change2$mx_num_diff <- (disease_life_table_list_sc[[index]]$mx * general_life_ta

    names(output_burden_change2)[names(output_burden_change2) == 'inc_num_bl'] <-
      paste('inc_num_bl', disease, sep = "_")
    names(output_burden_change2)[names(output_burden_change2) == 'inc_num_sc'] <-
      paste('inc_num_sc', disease, sep = "_")
    names(output_burden_change2)[names(output_burden_change2) == 'inc_num_diff'] <-
      paste('inc_num_diff', disease, sep = "_")
    names(output_burden_change2)[names(output_burden_change2) == 'mx_num_bl'] <-
      paste('mx_num_bl', disease, sep = "_")
    names(output_burden_change2)[names(output_burden_change2) == 'mx_num_sc'] <-
      paste('mx_num_sc', disease, sep = "_")
```

```
          names(output_burden_change2)[names(output_burden_change2) == 'mx_num_diff'] <-
            paste('mx_num_diff', disease, sep = "_")

          ## Bind all lists

          output_burden_sc <- cbind(output_burden_sc, td3)
          output_burden_sc <- cbind(output_burden_sc, td4)
          output_burden_sc$age_cohort <- age
          output_burden_sc <- cbind(output_burden_sc, output_burden_change2)

        }

        # cat(age, " - ", sex," - ",  disease," - ",  index, " - ", l_index,  "\n")
        index <- index + 1
      }

    }

    ## general_life_table_list_sc and general_life_table_list_bl (Lx)
    output_burden_lf_sc <- select(general_life_table_list_sc[[l_index]], c('Lx', 'Lwx'))
    names(output_burden_lf_sc)[names(output_burden_lf_sc) == 'Lx'] <- paste('Lx', "sc", sep = "_")
    names(output_burden_lf_sc)[names(output_burden_lf_sc) == 'Lwx'] <- paste('Lwx', "sc", sep = "_")

    output_burden_lf_bl <- select(general_life_table_list_bl[[l_index]], c('Lx', 'Lwx'))
    names(output_burden_lf_bl)[names(output_burden_lf_bl) == 'Lx'] <- paste('Lx', "bl", sep = "_")
    names(output_burden_lf_bl)[names(output_burden_lf_bl) == 'Lwx'] <- paste('Lwx', "bl", sep = "_")

    output_burden_lf_sc$Lx_diff <- general_life_table_list_bl[[l_index]]$Lx - general_life_table_list_s
    output_burden_lf_sc$Lwx_diff <- general_life_table_list_bl[[l_index]]$Lwx - general_life_table_list_

    output_burden_sc <- cbind(output_burden_sc, output_burden_lf_sc)
    output_burden_sc <- cbind(output_burden_sc, output_burden_lf_bl)

    output_burden[[l_index]] <- output_burden_sc
    l_index <- l_index + 1

  }
}

## Uncomment to check
# View(output_burden[[1]])
```

24) Combine all lists of outputs in a data frame to facilitate extraction of outcomes of interest and plotting.

```
#####Generate a data frame for all results and create function to get outcomes.

output_df <- plyr::ldply(output_burden, rbind)

# View(output_df)

#Remove variables that are not used in the generation of outputs. CHANGE THIS NAMES, TOO LONG

output_df <- subset(output_df, select = -c(incidence_disease_ihd_bl, incidence_disease_ihd_sc, incidence
                                           incidence_disease_colon_cancer_bl, incidence_disease_colon_ca
```

25) Plot outcomes for each age and sex cohort using created functions. Four functions are used: plot_output (plots outputs), grid_arrange_shared_legend and g_legend (places one label per page) and get_qualified_disease_name (replaces full name of disease)

```
output_dir = "output/graphs2"

i_outcome <- c("mx", "inc")
p_list_male <- list()
p_list_female <- list()
male_index <- 1
female_index <- 1
for (age in i_age_cohort){
  for (sex in i_sex) {
    for (outcome in i_outcome) {
      for (disease in i_disease){

        if (sex == "male" && disease == "breast_cancer"){
          # cat("\n")
        }else{

          p_index  <- plot_output(in_data = output_df, in_age = age, in_population = sex, in_outcomes =

          if (sex == "male"){

            p_list_male[[male_index]] <- p_index

            if (male_index %% 4 == 0 && male_index > 0){

              p1 <- p_list_male[[male_index - 3]] + theme(legend.position="none", axis.title.x = elemen
              p2 <- p_list_male[[male_index - 2]] + theme(legend.position="none", axis.title.x = elemen
              p3 <- p_list_male[[male_index - 1]] + theme(legend.position="none", axis.title.x = elemen
              p4 <- p_index + theme(legend.position="none", axis.title.x = element_blank(),  axis.title
```

```
            jpeg(paste0(output_dir, "/", paste(age, sex, outcome, sep="_"), ".jpeg"))
            grid_arrange_shared_legend (p1, p2, p3, p4, ncol = 2, nrow = 2, mainTitle = paste(ifelse(
                                        mainLeft = 'Cases', mainBottom = 'Age')
            dev.off()

          }

          male_index <- male_index + 1

        }

        if (sex == "female" && female_index > 0){
          p_list_female[[female_index]] <- p_index

          if (female_index %% 5 == 0){
            p1 <- p_list_female[[female_index - 4]] + theme(legend.position="none", axis.title.x = ele
            p2 <- p_list_female[[female_index - 3]] + theme(legend.position="none", axis.title.x = ele
            p3 <- p_list_female[[female_index - 2]] + theme(legend.position="none", axis.title.x = ele
            p4 <- p_list_female[[female_index - 1]] + theme(legend.position="none", axis.title.x = ele
            p5 <- p_index + theme(legend.position="none", axis.title.x = element_blank(),  axis.title

            jpeg(paste0(output_dir, "/", paste(age, sex, outcome, sep="_"), ".jpeg"))
            grid_arrange_shared_legend (p1, p2, p3, p4, p5, ncol = 2, nrow = 3, mainTitle = paste(ifel
            dev.off()

          }
          female_index <- female_index + 1
        }

      }
    }
  }
}




## Loop to include graphs in the document

graphs_doc <- list()
index <- 1
```

```
for (age in i_age_cohort) {
  for (sex in i_sex)  {
    for (outcome in i_outcome) {

      graphs_doc [[index]] <- c(paste(output_dir, "/",age,"_",sex,"_", outcome,".jpeg", sep = ""))
      knitr::include_graphics(graphs_doc [[index]])

      index <- index + 1
    }
  }
}
```

26) Add up all outcomes per year of simulation with generated function. BETTER TO HAVE A LOOP, I TRIED BY LOOPING IN THE FUNCITON ARGUMENT I_OUTCOMES, BUT THIS DID NOT WORK. THEN, IF LOOP HERE ALSO IN NEXT CODE.

```
## Try binding function

aggregate_frame_males <- list()
aggregate_frame_females <- list()

index <- 1

for (outcome in i_outcome) {
  for (disease in i_disease) {

    aggregate_frame_males[[index]] <- gen_aggregate(in_data = output_df, in_cohorts = 16, in_population

    aggregate_frame_females[[index]] <- gen_aggregate(in_data = output_df, in_cohorts = 16, in_populati

    # Remove non-numeric columns starting with age and sex
    aggregate_frame_males[[index]] <- aggregate_frame_males[[index]] %>% select(-starts_with("age"), -s

    aggregate_frame_females[[index]] <- aggregate_frame_females[[index]] %>% select(-starts_with("age")

    index <- index + 1
  }
}


## Loop for life years (Lx) and health adjusted life years (Lwx) to then add to total aggregated data.

## RUN ANOTHER LOOP FOR LX AND LWX
```

```r
i_outcome2 <- c("Lx", "Lwx")

aggregate_frame_males2 <- list()
aggregate_frame_females2 <- list()

for (i in i_outcome2){

  aggregate_frame_males2[[i]] <- gen_aggregate(in_data = output_df, in_cohorts = 16, in_population = "ma

  aggregate_frame_females2[[i]] <- gen_aggregate(in_data = output_df, in_cohorts = 16, in_population = "

  aggregate_frame_males2[[i]] <- aggregate_frame_males2[[i]] %>% select(-starts_with("age"), -starts_wi

  aggregate_frame_females2[[i]] <- aggregate_frame_females2[[i]] %>% select(-starts_with("age"), -starts

}

## Uncomment to check data frames (list correspond to a disease and outcome combination, for example,
## incidence breast cancer)

# View(aggregate_frame_females[[2]])

## Transform list to data frame (this is not working after the first disease)

aggregate_frame_females <- do.call(cbind, aggregate_frame_females)
aggregate_frame_males <- do.call(cbind, aggregate_frame_males)
aggregate_frame_females2 <- do.call(cbind, aggregate_frame_females2)
aggregate_frame_males2 <- do.call(cbind, aggregate_frame_males2)

View(aggregate_frame_females2)

## Drop ending of variables names _males/_females to add up all outcomes in next step.

names(aggregate_frame_females) = gsub(pattern = "_female", replacement = "", x = names(aggregate_frame_

names(aggregate_frame_males) = gsub(pattern = "_male", replacement = "", x = names(aggregate_frame_males

names(aggregate_frame_females2) = gsub(pattern = "_female", replacement = "", x = names(aggregate_frame_

names(aggregate_frame_males2) = gsub(pattern = "_male", replacement = "", x = names(aggregate_frame_mal
```

```
## Uncomment to check that the code is dropping the male/female ending.
# View(aggregate_frame_males)

## Create a copy of aggregate_frame_females.
total_aggr1 <- aggregate_frame_females
## Add aggregate_frame_males values to it
for (i in 1:ncol(aggregate_frame_females)){
  total_aggr1[i] <- total_aggr1[i] + aggregate_frame_males[i]
}

## Add data frames 2 with life years (check adds Lx and Lwx at the beginning of the variable name)
total_aggr2 <- aggregate_frame_females2
## Add aggregate_frame_males values to it
for (i in 1:ncol(aggregate_frame_females2)){
  total_aggr2[i] <- total_aggr2[i] + aggregate_frame_males2[i]
}

## Combine data frames

total_aggr <- cbind.data.frame(total_aggr1, total_aggr2)

total_aggr$sim_year <- seq.int(nrow(total_aggr))


## Uncomment to see total data frame
View(total_aggr)

## Test that total_aggr is adding males and females dataframes
#
# (aggregate_frame_females2$Lx.Lx_bl_22 + aggregate_frame_males2$Lx.Lx_bl_22) -total_aggr$Lx.Lx_bl_22
```

27) Plots using total aggregated change in burden. This graphs are not very good for life years and health adjusted life years as the difference between baseline and scenario is very small relative to the total, hence the graphs show almost no change.

```
####This plot has to be customised to in_outcomes, here, only totals shown, but specifications are up t

####[] is used here to indicate the number of simulation years into the future.
####Disease outcomes has to be changed to the outcome of interest

#### Test code with loops for aggregated outcomes diseases burden. NOT WORKING.

### Compare with loops for age and sex cohort outcomes.
```

```
p_aggr_list <- list()
index <- 1

for (outcome in i_outcome) {
  for (disease in i_disease) {
    # outcome <- i_outcome[1]
    # disease <- i_disease[1]

    p_aggr_list_index <- ggplot(total_aggr[1:79,], aes(x = total_aggr[["sim_year"]])) +

      geom_line(mapping = aes(y = total_aggr[[paste("total", outcome, "num_bl", disease, sep = "_")]],
      theme_classic() +
      geom_hline(yintercept=0, linetype="dashed", color = "black") +
      geom_line(mapping = aes(y = total_aggr[[paste("total", outcome, "num_sc", disease, sep = "_")]],
      geom_line(mapping = aes(y = total_aggr[[paste("total", outcome, "num_diff", disease, sep = "_")]]
      xlab ("Simulation years") + ylab ("Cases") + labs (title = paste(disease, outcome)) +
      theme(plot.title = element_text(hjust = 0.5, size = 12)) +
      scale_color_discrete(name = paste(""), labels = c("Baseline", "Difference", "Scenario")) +
      theme(plot.title = element_text(hjust = 0.5))
    p_aggr_list[[index]] <- p_aggr_list_index
    index <- index + 1


  }
}

index <- 1

interpolation_index <- 1
for (outcome in i_outcome) {
  for (disease in i_disease) {
    file_name = paste("output/graphs2/", "Aggregated Outcomes", outcome, disease, ".jpeg", sep=" ")
    jpeg(file_name)
    print(p_aggr_list[[index]])
    index <- index + 1
    dev.off()
  }
}


p_aggregated <- do.call(marrangeGrob, list(grobs=p_aggr_list, nrow = 2, ncol = 2))
p_aggregated
```

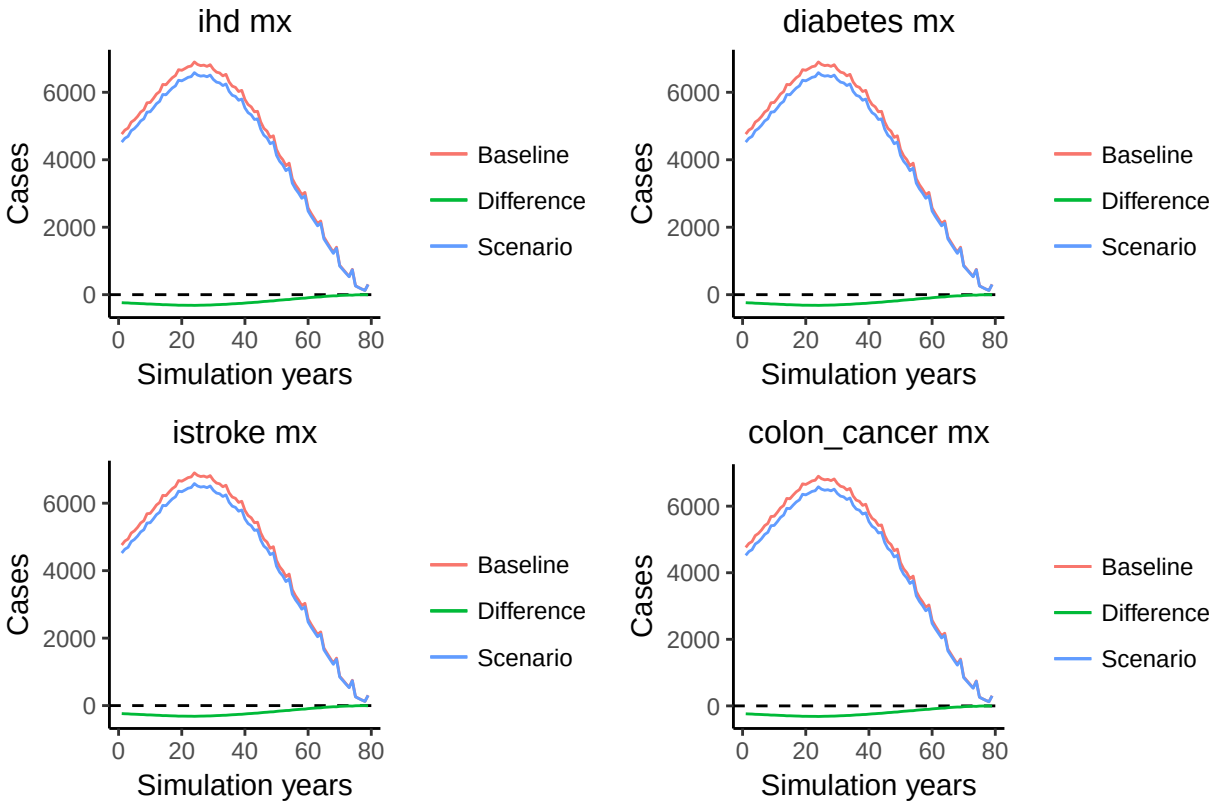

Figure 6: FALSE

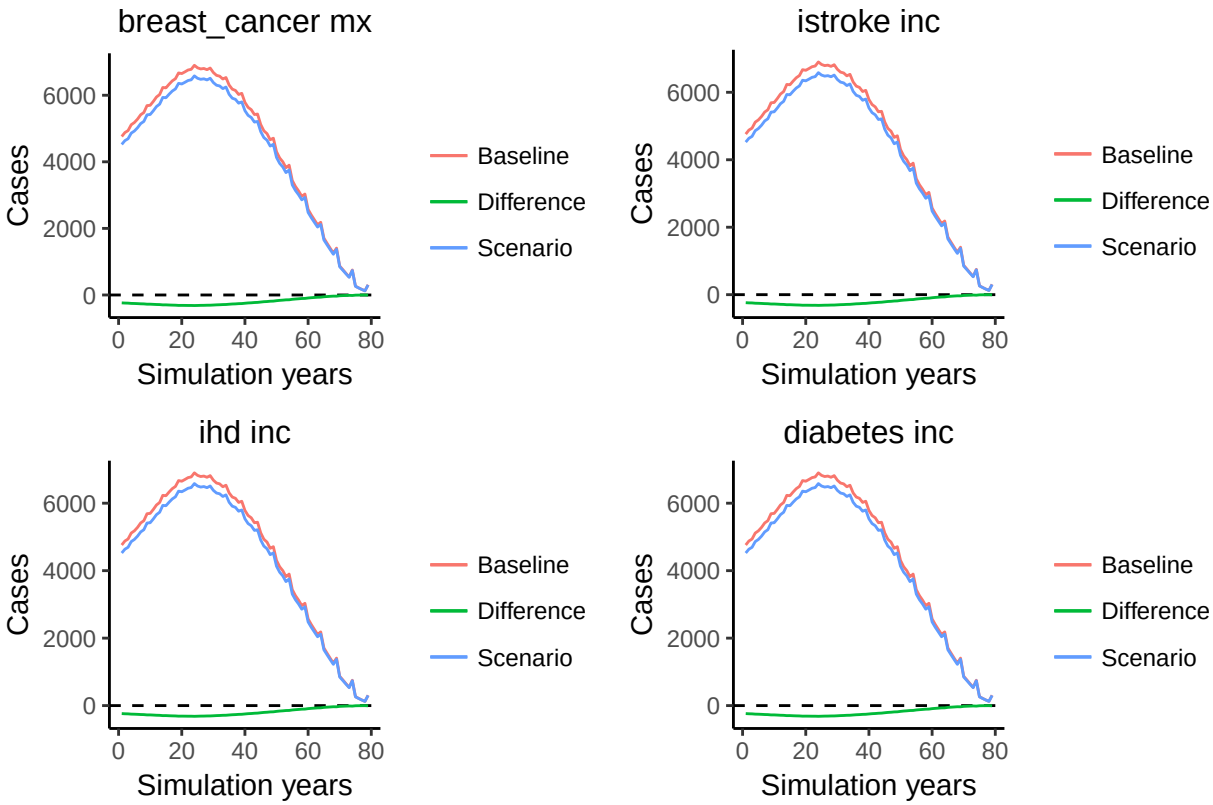

Figure 7: FALSE

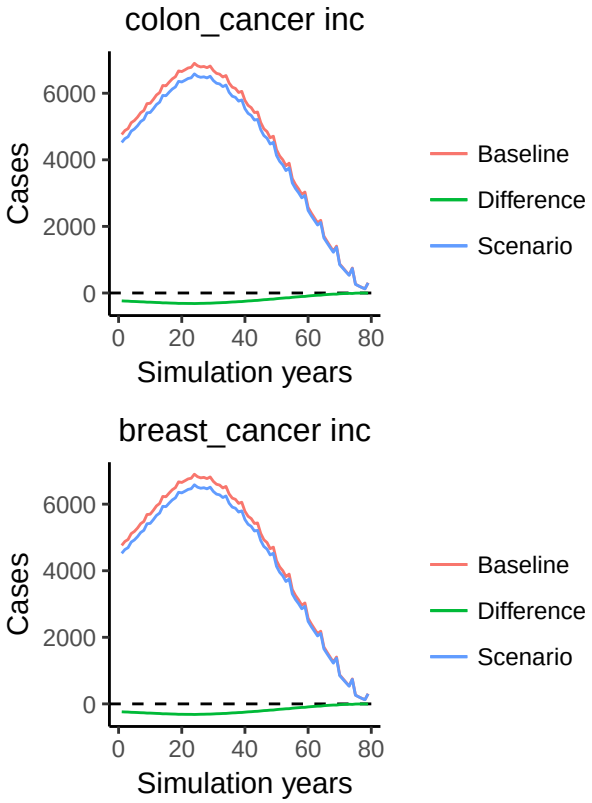

Figure 8: FALSE

3 Comments

3.1 Road injuries in the PMsLT

The disease model used in each of the disease life table is not directly applicable to road injuries, however, similar concept can be follow. Firstly, changes in road fatalities impact on the overall mortality rate, hence, by knowing the road fatality rates for baseline and scenarios, we will be able to incorporate changes to mortality attributable to road fatalities. For road injuries, methods developed by Kavi Bhalla and Marko Tanio (REFS) that derive the average YLD attributable to life long and short term injuries can be applied to derive the change in total YLDs (CHECK THAT THESE WERE DEVELOPED AS INCIDENCE YLDs).MT's methods assumes that injuries do not reduce the life expectancy of the injured person.

References

Barendregt, J.J., and J.L. Veerman. 2010. "Categorical Versus Continuous Risk Factors and the Calculation of Potential Impact Fractions." Journal Article. *J Epidemiol Community Health* 64 (3): 209–12.

doi:10.1136/jech.2009.090274.

Blakely, T., L. J. Cobiac, C. L. Cleghorn, A. L. Pearson, F. S. Deen, G. Kvizhinadze, N. Nghiem, M. McLeod, and N. Wilson. 2015. "Health, Health Inequality, and Cost Impacts of Annual Increases in Tobacco Tax: Multi-state Life Table Modeling in New Zealand." Journal Article. *PLoS Med* 12. doi:10.1371/journal.pmed.1001856.

Briggs, Adam, Peter Scarborough, and Adrian Smith. 2016. "Modelling in Public Health." Book Section. In *Public Health Intelligence: Issues of Measure and Method*, edited by Krishna Regmi and Ivan Gee, 67–90. Cham: Springer International Publishing. doi:10.1007/978-3-319-28326-5_4.

Cobiac, L.J., T. Vos, and J.J. Barendregt. 2009. "Cost-Effectiveness of Interventions to Promote Physical Activity: A Modelling Study." Journal Article. *Plos Med* 6 (7): e1000110–e1000110. doi:10.1371/journal.pmed.1000110.

Gold, Marthe R., David Stevenson, and Dennis G. Fryback. 2002. "HALYs and Qalys and Dalys, Oh My: Similarities and Differences in Summary Measures of Population Health." Journal Article. *Annu Rev Public Health* 23 (1): 115–34. doi:doi:10.1146/annurev.publhealth.23.100901.140513.

Murray, Christopher J. L., Majid Ezzati, Abraham D. Flaxman, Stephen Lim, Rafael Lozano, Catherine Michaud, Mohsen Naghavi, et al. 2012. "GBD 2010: Design, Definitions, and Metrics." Journal Article. *The Lancet* 380 (9859): 2063–6. doi:10.1016/S0140-6736(12)61899-6.

Roux, L., M. Pratt, and T. O. Tengs. 2008. "Cost Effectiveness of Community-Based Physical Activity Interventions." Journal Article. *Am J Prev Med* 35. doi:10.1016/j.amepre.2008.06.040.

Vos, T., R. Carter, J. J Barendregt, Mihalopoulos C., JL. Veerman, A. Magnus, L. Cobiac, Bertram MY., and AL. Wallace. 2010. "Assessing Cost-Effectiveness in Prevention (Ace-Prevention): Final Report." Report.