DOHMH Roadmap: DALY Estimates

Ryan Quan (rcq2102)

April 17, 2015

Contents

Code

Preliminaries

First, we load our dependencies into the R environment.

```
library("plyr")
library("dplyr")
library("reshape2")
library("magrittr")
library("ggplot2")
library("grid")
library("scales")
dir.create("results")
dir.create("data")
```

Next, we define a set of functions that we will be using for our analysis. Details on the parameters and return values for each function can be found in the comment blocks below:

```
readData <- function(url) {
    ## Reads CSV data from input URL string
    filename <- tail(unlist(strsplit(url, "/")), 1)
    filepath <- paste("data", "/", filename, sep="")
```

```
    if (!file.exists(filepath)) {
    download.file(url, filename, method="curl")
    }
    data <- read.csv(filepath, stringsAsFactors=FALSE)
    return(data)
}

assignAgeGroup <- function(ageVar) {
    ## logic for childhood, teenage, young adult, adult, and later-life age groups
    if (ageVar %in% c("Under 5 years", "5-14 years")) {
        return("00-14")
    } else if (ageVar %in% c("15-19 years", "20-24 years")) {
        return("15-24")
    } else if (ageVar %in% c("25-29 years", "30-34 years", "35-39 years", "40-44 years")) {
        return("25-44")
    } else if (ageVar %in% c("45-49 years", "50-54 years", "55-59 years", "60-64 years")) {
        return("45-64")
    } else if (ageVar %in% c("65-69 years", "70+ years")) {
        return("65+")
    } else {
        return("")
    }
}

addAgeGroup <- function(data, ageVar="age_name") {
    ## replaces age grouping in current data.frame to childhood, teenage, YA, adult, later-life
    ## Args:
    ##      data: data.frame object
    ##      ageVar: string denoting the column of ages to be replaced
    ## Returns:
    ##      data: data.frame object with new age groupings
    ageGroup <- vector(length=nrow(data))
    for (i in 1:nrow(data)) {
        ageGroup[i] <- assignAgeGroup(as.vector(data[i, ageVar]))
    }
    data$ageGroup <- ageGroup
    return(data)
}

preprocessGBD <- function(data) {
    ## extracts YLD and YLL rates from 2010 Global Burden of Disase data
    ## Args:
    ##      data: GBD dataset downloaded from the web
    ## Returns:
    ##      data: a pre-processed 2010 GBD dataset
    data %<>%
        ## filter out unnecessary variables
        select(-c(pc_mean, pc_upper, pc_lower)) %>%
        filter(year == 2010) %>%
        filter(sex %in% c("Females", "Males")) %>%
        ## extract only YLD and YLL rates
        filter(measure %in% c("yll", "yld")) %>%
        ## create long-form dataset
        melt(measure.vars=c("nm_mean", "nm_upper", "nm_lower", "rt_mean", "rt_upper", "rt_lower")) %>%
```

```
        ## create wide-form dataset with national YLD/YLL rates
        dcast(cause_name + age_name + sex ~ measure + variable, value.var="value") %>%
        ## age group manipulations
        addAgeGroup("age_name") %>%
        filter(ageGroup != "") %>%
        select(-age_name) %>%
        ## averaging YLD/YLL rates with respect to new age groupings
        group_by(cause_name, sex, ageGroup) %>%
        summarise_each(funs(mean))
    return(data)
}

getDiseaseIndex <- function(diseaseName, data) {
    ## searches disease index and returns indices of the first match
    ## Args:
    ##      diseaseName: string vector denoting diseases of interest
    ##      data: data.frame to be searched
    ## Returns:
    ##      indices of the first string match
    index <- grep(diseaseName, data$cause_name)
    pattern <- unique(data$cause_name[index])[1]
    return(which(data$cause_name == pattern))
}

subsetDataByDisease <- function(diseaseName, data) {
    ## subsets data frame from first string match
    index <- getDiseaseIndex(diseaseName, data)
    return(data[index, ])
}
```

This function contains the logic from the Michaud, 2006 study.

```
calculateMichaudYLD <- function(checkRatio, yldyllRatio, nationalYLD, nycPop, nycYLL) {
    ## calculates YLDs based on the 2006 Michaud study
    ## Args:
    ##      checkRatio: numeric. National YLD:YLL ratio to check if > 10 or < 10
    ##      yldyllRatio: numeric. National YLD:YLL ratio to evaluate
    ##      nationalYLD: numeric. National YLD rate
    ##      nycPop: numeric. NYC Population
    ##      nycYLL: numeric. NYC YLL
    ## Returns:
    ##      nycYLD: New York City YLD estimate
    nycYLDLogic <- (checkRatio >= 5 | is.na(checkRatio) | is.infinite(checkRatio) | is.na(nycYLL))
    nycYLD <- ifelse(nycYLDLogic, nationalYLD * (nycPop / 100000), yldyllRatio * nycYLL)
    return(nycYLD)
}
```

This function implements prevalence-based YLD estimates.

```
calculatePrevalenceYLD <- function(nycPrevalence) {
   ## calculates prevalence-based YLD estimates from 2010 GBD Study
    ## Args:
    ##      nycPrevalence: data.frame. NYC prevalence data with associated disability weights
```

```
    ## Returns:
    ##      nycYLD: data.frame. NYC YLD estimates.
    nycYLD <- nycPrevalence %>%
        mutate(yld = prevalence * dependence_rate * dw_estimate,
               yld_upper = prevalence * dependence_rate * dw_upper,
               yld_lower = prevalence * dependence_rate * dw_lower)
    return(nycYLD)
}

calculateYLL <- function(mortalityData) {
    ## calculates YLLs from mortality data
    nycYLL <- mortalityData %>%
        mutate(le = sle - mean_age,
               yll = mortality * (1 - exp((-0.03 * le))) / 0.03)
    return(nycYLL)
}

calculatePrevalenceDALY <- function(diseaseName, nycYLL, nycYLD) {
    ## calculates DALYs using prevalence-based YLDs from the 2010 GBD study
    ## Args:
    ##      diseaseName: chr. The disease of interest.
    ##      nycYLL: data.frame. New York City YLL estimates
    ##      nycYLD: data.frame. New York City YLD estimates
    ## Returns:
    ##      dalys: data.frame. New York City DALY estimates
        diseaseYLL <- subsetDataByDisease(diseaseName, nycYLL)
        nycYLD <- subsetDataByDisease(diseaseName, nycYLD)
        dalys <- diseaseYLL %>%
            group_by(cause_name, sex) %>%
            summarize(yll = sum(yll)) %>%
            join(nycYLD, c("cause_name", "sex"), type = "right") %>%
            ungroup() %>%
            mutate(daly = ifelse(is.na(yll), 0 + yld, yll + yld),
                   daly_upper = ifelse(is.na(yll), 0 + yld_upper, yll + yld_upper),
                   daly_lower = ifelse(is.na(yll), 0 + yld_lower, yll + yld_lower))
        return(dalys)
}

calculateDALY <- function(diseaseName, population, nycYLL, nycYLD=NULL, nationalRates=NULL) {
    ## workhorse function to calculate DALY scores for specified disease using either
    ## prevalence-based YLD estimates or the Michaud approach using national YLD/YLL rates
    diseaseYLL <- subsetDataByDisease(diseaseName, nycYLL)
    if (!is.null(nycYLD) & !is.null(nationalRates)) {
        stop("You cannot provide values to both nycYLD and nationalRates parameters.")

    } else if (!is.null(nycYLD)) {
        nycYLD <- subsetDataByDisease(diseaseName, nycYLD)
        dalys <- calculatePrevalenceDALY(diseaseName, nycYLL, nycYLD)
        return(dalys)

    } else if (!is.null(nationalRates)) {
        ## subset datasets for specified disease
        diseaseRates <- subsetDataByDisease(diseaseName, nationalRates)
```

```
        ## if disease not found in gbdData, return YLL data as DALYs
        if (nrow(diseaseRates) == 0) {
            dalys <- diseaseYLL %>%
                group_by(cause_name, sex) %>%
                summarize(yll = sum(yll),
                          daly = sum(yll))
            return(dalys)
        }
        ## compute national YLD:YLL ratio and join to NYC YLL and population data by age, sex
        dalys <- diseaseRates %>%
            ## compute national YLD:YLL ratio
            mutate(yldyll_ratio_mean = yld_nm_mean / yll_nm_mean,
                   yldyll_ratio_upper = yld_nm_upper / yll_nm_mean,
                   yldyll_ratio_lower = yld_nm_lower / yll_nm_mean) %>%
            # join tables
            join(population, by=c("ageGroup", "sex")) %>%
            join(diseaseYLL, by=c("cause_name", "ageGroup", "sex")) %>%
            ## estimate YLDs using Michaud logic
            mutate(yld = calculateMichaudYLD(yldyll_ratio_mean, yldyll_ratio_mean, yld_rt_mean, populati
                   yld_upper = calculateMichaudYLD(yldyll_ratio_mean, yldyll_ratio_upper, yld_rt_upper,
                   yld_lower = calculateMichaudYLD(yldyll_ratio_mean, yldyll_ratio_lower, yld_rt_lower,
            ## collapse age groups
            group_by(cause_name, sex) %>%
            summarise_each(funs(sum(., na.rm=TRUE)), -c(cause_name, sex, ageGroup)) %>%
            ## calculate DALY estimates with lower and upper bounds
            mutate(daly = yll + yld,
                   daly_upper = yll + yld_upper,
                   daly_lower = yll + yld_lower) %>%
            select(cause_name, sex, yll, yld, yld_upper, yld_lower, daly, daly_upper, daly_lower)
        return(dalys)
    }
}

segmentDALY <- function(dalyObj, strata) {
    ## helper function to subset DALY data
    if (strata == "total") {
        dalyObj %>% group_by(cause_name) %>% summarise_each(funs(sum), -c(sex)) %>% arrange(desc(daly))
    } else if (strata == "male") {
        dalyObj %>% filter(sex == "Male") %>% arrange(desc(daly))
    } else if (strata == "female") {
        dalyObj %>% filter(sex == "Female") %>% arrange(desc(daly))
    }
}

# Multiple plot function
#
# ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects)
# - cols:   Number of columns in layout
# - layout: A matrix specifying the layout. If present, 'cols' is ignored.
#
# If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE),
# then plot 1 will go in the upper left, 2 will go in the upper right, and
# 3 will go all the way across the bottom.
#
```

```
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  library(grid)

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)

  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }

 if (numPlots==1) {
    print(plots[[1]])

  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}

plotDALY <- function(data, title, stackedBar=FALSE) {
    ## plot function for DALY object
    if (stackedBar) {
        meltedData <- melt(data, id.vars="cause_name", measure.vars=c("yll", "yld"), value.name="daly")
        ggplot(meltedData, aes(x=reorder(cause_name, daly, FUN=sum, na.rm=TRUE), y=daly, fill=variable)
            geom_bar(stat="identity") +
            ggtitle(title) +
            ylab("Disability-Adjusted Life Years (DALYs)") + xlab("Causes") +
            scale_y_continuous(breaks=seq(0, max(data$daly_upper, na.rm=TRUE), by=100000), labels=comma
            scale_fill_brewer() +
            coord_flip() +
            theme_bw()
    } else {
        limits <- aes(ymin=daly_lower, ymax=daly_upper)
        ggplot(data, aes(x=reorder(cause_name, daly), y=daly)) +
            geom_pointrange(limits) +
            ggtitle(title) +
            ylab("Disability-Adjusted Life Years (DALYs)") + xlab("Causes") +
```

```
            scale_y_continuous(breaks=seq(0, max(data$daly_upper, na.rm=TRUE), by=100000), labels=comma)
            coord_flip() +
            theme_bw()
    }
}
```

Reading in the Data

To make our analysis reproducible, we download the 2010 Global Burden of Disease data straight from the source using the `readData()` function.

```
url <- "http://ghdx.healthdata.org/sites/default/files/record-attached-files/IHME_USA_GBD_2010_RESULTS_
cause <- readData(url) %>%
    preprocessGBD()

url <- "http://ghdx.healthdata.org/sites/default/files/record-attached-files/IHME_USA_GBD_2010_RESULTS_
risk <- readData(url) %>%
    rename(cause_name = risk_name) %>%
    preprocessGBD()
```

Next, we read in the mortality, population, and prevalence data provided by NYCDOHMH.

```
mortality <- read.csv("data/2013_nyc_mortality.csv", stringsAsFactors=FALSE)
population <- read.csv("data/2013_nyc_population.csv", stringsAsFactors=FALSE)
prevalence <- read.csv("data/2013_nyc_prevalence.csv", stringsAsFactors=FALSE)
```

Data Preparation

We pre-process the national YLD/YLL rates by substituting values for `cause_name` in order to match the indices of the other datasets. This will allow us to merge datasets using `cause_name` as the key. We also write out the resulting dataset for inspection.

```
nationalRates <- rbind(cause, risk) %>%
    ungroup() %>%
    mutate(sex = ifelse(sex == "Females", "Female", "Male")) %>%
    mutate(cause_name = ifelse(cause_name == "Road injury", "Motor vehicle accidents", cause_name),
           cause_name = ifelse(cause_name == "Trachea, bronchus, and lung cancers", "Lung cancer", cause
    arrange(cause_name)
write.csv(nationalRates, "results/national_yldyll_rates.csv")
```

Next, we pre-process the NYC mortality and calculate the YLLs for each disease by age, sex, and race. For the analysis, we only use YLLs stratified by age and sex.

```
nycYLL <- calculateYLL(mortality)
write.csv(nycYLL, "results/nyc_yll_by_age_sex_race.csv")

nycYLL %<>%
    group_by(cause_name, sex, ageGroup) %>%
    summarize(yll = sum(yll))
write.csv(nycYLL, "results/nyc_yll_by_age_sex.csv")
```

We calculate YLDs for each condition using NYC prevalence data, which also contains the associated disability weights for each disease. To capture the level of uncertainty around disability weights, we include the upper and lower bounds of the resulting YLDs in the output.

```
nycYLD <- calculatePrevalenceYLD(prevalence)
write.csv(nycYLD, "results/nyc_yld_by_age_sex.csv")

nycYLD %<>%
    group_by(cause_name, sex) %>%
    summarize(yld = sum(yld, na.rm=TRUE),
              yld_upper = sum(yld_upper, na.rm=TRUE),
              yld_lower = sum(yld_lower, na.rm=TRUE))
write.csv(nycYLD, "results/nyc_yld_by_sex.csv")
```

DALY Estimation

Michaud YLD Approach

This section contains an implementation of the Michaud approach described in the above methods section. We first create a search index containing all the disease conditions of interest.

```
## create a search index
disease <- unique(c(nycYLL$cause_name, nycYLD$cause_name))
drug <- c("Amphetamine", "Heroin", "Cocaine", "Cannabis")
mental <- c("Major depressive disorder", "Anxiety", "Bipolar")
index <- unique(c(disease, drug, mental))
```

This search index is then fed through the `calculateDALY` workhorse function to estimate DALYs for each disease condition. The result is a `data.frame` object containing the following columns: `cause_name`, `sex`, `yll`, `yld`, `yld_upper`, `yld_lower`, `daly`, `daly_upper`, `daly_lower`.

```
michaudDALY <- lapply(index, calculateDALY, population, nycYLL=nycYLL, nationalRates=nationalRates)
michaudDALY <- do.call(rbind.fill, michaudDALY)
write.csv(michaudDALY, "results/nyc_daly_michaud.csv")
```

Prevalence-Based YLD Approach

Similar to the section, we implement the prevalence-based YLD approach here using the same search index.

```
prevalenceDALY <- lapply(index, calculateDALY, population, nycYLL=nycYLL, nycYLD=nycYLD)
prevalenceDALY <- do.call(rbind.fill, prevalenceDALY)
write.csv(prevalenceDALY, "results/nyc_daly_prevalence.csv")
```

Results

Michaud YLD Approach

Raw results for this approach can be found under the `results` directory under the filename `nyc_daly_michaud.csv`. The file can be opened in Excel and manipulated with a pivot table for aggregation and stratification purposes.

2013 NYC DALY Estimates, Total

```
michaudTotal <- segmentDALY(michaudDALY, strata="total")
knitr::kable(michaudTotal, digits=0)
```

```
plotDALY(michaudTotal, "Leading Causes of DALYs, NYC 2013")
plotDALY(michaudTotal, "Leading Causes of DALYs, NYC 2013", stackedBar=TRUE)
```

- `Diabetes mellitus` is the leading cause of disease in 2013, but has a wide range of uncertainty
- Disaggregated drug use disorders ranked relatively low, particuarly for non-alcohol-related substances
- `Major depressive disorder` just missed the top 10 cutoff

2013 NYC DALY Estimates, Male

```
michaudMale <- segmentDALY(michaudDALY, strata="male")
knitr::kable(michaudMale, digits=0)
```

```
plotDALY(michaudMale, "Leading Causes of DALYs in Males, NYC 2013")
plotDALY(michaudMale, "Leading Causes of DALYs in Males, NYC 2013", stackedBar=TRUE)
```

- `Alcohol use disorders` rises to the #4 slot
- `Homicide` and accidental deaths such as `poisonings` and `motor vehicle accidents` rise in rankings

2013 NYC DALY Estimates, Female

```
michaudFemale <- segmentDALY(michaudDALY, strata="female")
knitr::kable(michaudFemale, digits=0)
```

```
plotDALY(michaudFemale, "Leading Causes of DALYs in Females, NYC 2013")
plotDALY(michaudFemale, "Leading Causes of DALYs in Females, NYC 2013", stackedBar=TRUE)
```

- `Breast cancer` makes the top 3
- `Alzheimer's disease and other dementias` ranks very high
- Drug-related disorders get pushed to the bottom

Prevalence-Based YLD Approach

Raw results for this approach can be found under the `results` directory under the filename `nyc_daly_prevalence.csv`. The file can be opened in Excel and manipulated with a pivot table for aggregation and stratification purposes.

2013 NYC DALY Estimates, Total

```
prevalenceTotal <- segmentDALY(prevalenceDALY, strata="total")
knitr::kable(prevalenceTotal, digits=0)
```

- `Major depressive disorder` ranks number one, beating out the number two slot by almost twice the number of DALYs However, DALY estimates appear to be unstable, taking a wide range of possible values.

- Not enough information to calculate DALY estimates for `sedative use`, `stimulant use`, `tranquilizer use`.

```
plotDALY(prevalenceTotal, "Leading Causes of DALYs, NYC 2013")
plotDALY(prevalenceTotal, "Leading Causes of DALYs, NYC 2013", stackedBar=TRUE)
```

2013 NYC DALY Estimates, Male

```
prevalenceMale <- segmentDALY(prevalenceDALY, strata="male")
knitr::kable(prevalenceMale, digits=0)
```

```
plotDALY(prevalenceMale, "Leading Causes of DALYs in Males, NYC 2013")
plotDALY(prevalenceMale, "Leading Causes of DALYs in Males, NYC 2013", stackedBar=TRUE)
```

- `Alcohol use disorders` rises in proportion to `major depressive disorder`

2013 NYC DALY Estimates, Female

```
prevalenceFemale <- segmentDALY(prevalenceDALY, strata="female")
knitr::kable(prevalenceFemale, digits=0)
```

```
plotDALY(prevalenceFemale, "Leading Causes of DALYs in Females, NYC 2013")
plotDALY(prevalenceFemale, "Leading Causes of DALYs in Females, NYC 2013", stackedBar=TRUE)
```

Michaud YLDs vs. Prevalence-Based YLDs: Side-by-Side Comparison

Total

```
multiplot(plotDALY(michaudTotal, "Michaud YLDs"), plotDALY(prevalenceTotal, "Prevalence-Based YLDs"))
```

Male

```
multiplot(plotDALY(michaudMale, "Michaud YLDs"), plotDALY(prevalenceMale, "Prevalence-Based YLDs"))
```

Female

```
multiplot(plotDALY(michaudFemale, "Michaud YLDs"), plotDALY(prevalenceFemale, "Prevalence-Based YLDs"))
```

Disease Conditions with Small Sample Sizes

```
prevalence[prevalence$small_sample == "yes", c("cause_name", "sequlae", "sex", "age")]
```

Session Info

```
sessionInfo()
```