README\_R\_functions

Hannah Allen

April 19, 2017

Table of Contents

# Function names listed below

# age.R

**Purpose**: Calculate age of patient in year, month, and day of life for any dataframe with patient's medical record number, date of birth, and date on therapy.

**What you need to know**: This function requires the lubridate package and uses the interval function in the lubridate package to get age in years and months. If you do not have lubridate installed then enter the following to install and library the package

install.packages("lubridate")  
library(lubridate)

If the data frame you are using does not have date of birth then you will need to add that in before using age.R. The subset\_dopkt.R function will add date of birth to your data frame if you have patient medical record number and date on therapy.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name\_calculated is the name of the data frame you want the day of life, month, and year calculated age values to show up in and table\_name is the name of the data frame you want to calculate day of life, month, and year age values from.

source("https://raw.githubusercontent.com/borumlab/functions/master/age.R")  
table\_name\_calculated <- age(table\_name)

# age\_initiation.R

**Purpose**: calculate age at initiation and subset patients who are less than your selected age

**What you need to know**: This function uses the table demographics\_id\_calculated and the column headers called "DOB" and "PKT\_INITIATED\_DATE". This function gives you the ability to only include patients who were less than a specified age at baseline.

**How to use**: Paste the lines below into your R code and modify per criteria where demographics\_id\_calculated is the name of the data frame. Age in years is the maximum age of patients you want included at baseline.

source("https://raw.githubusercontent.com/borumlab/functions/master/age\_initiation.R")  
demographics\_id\_calculated <- age\_initiation(age in years)

# baseline\_bmi.R

**Purpose**: Subsets a population if they have a baseline BMI.

**What you need to know**: This function uses the table called anthropometrics\_id\_research and the column headers called "BMI\_DAY", 'DAY\_TYPE", and "MRNUMBER". This function will only create a list of medical record numbers for patients that have a BMI value in BMI\_DAY for DAY\_TYPE equal to 1 (meaning baseline).

**How to use**: Paste the lines below into your R code and modify per criteria where baseline\_list\_bmi is the name of the data frame with the list of medical record numbers that have bmi values at baseline.

source("https://raw.githubusercontent.com/borumlab/functions/master/baseline\_bmi.R")  
baseline\_list\_bmi <- baseline\_bmi()

# therapy\_foodomics.R

**Purpose**: Subsets a population if they have a baseline foodomics.

**What you need to know**: This function uses the table called foodomics\_daily\_diet\_research and the column headers called "MRNUMBER". This function will only create a list of medical record numbers for patients that have a medical record number in this table (meaning the patient has foodomics data).

**How to use**: Paste the lines below into your R code and modify per criteria where baseline\_list\_foodomics is the name of the data frame with the list of medical record numbers that have foodomics values at baseline.

source("https://raw.githubusercontent.com/borumlab/functions/master/therapy\_foodomics.R")  
therapy\_list\_food <- therapy\_foodomics()

# baseline\_meds.R

**Purpose**: Subsets a population if they have a Med Load greater than a value x at baseline.

**What you need to know**: This function uses the table called med\_data\_id\_research and the column headers "MED\_LOAD\_DAY", "DAY\_TYPE", and "MRNUMBER". This function will only create a list of medical record numbers for patients that have a Med Load value in MED\_LOAD\_DAY for DAY\_TYPE equal to 1 (meaning baseline) and the Med Load value must be greater than x.

**How to use**: Paste the lines below into your R code and modify per criteria where baseline\_list\_meds is the name of the data frame with the list of medical record numbers that have Med Load values at baseline greater than x and x is the value you want Med Load to be greater than (e.g. 0).

source("https://raw.githubusercontent.com/borumlab/functions/master/baseline\_meds.R")  
baseline\_list\_meds <- baseline\_meds(x)

# baseline\_sz.R

**Purpose**: Subsets a population if they have a Seizure Load greater than a value x at baseline

**What you need to know**: This function uses the table called seizure\_data\_id\_research and the column headers "SEIZURE\_LOAD\_DAY", "DAY\_TYPE", and "MRNUMBER". This function will only create a list of medical record numbers for patients that have a Seizure Load value in SEIZURE\_LOAD\_DAY for DAY\_TYPE equal to 1 (meaning baseline) and the Seizure Load value must be greater than x.

**How to use**:Paste the lines below into your R code and modify per criteria where baseline\_list\_seizure is the name of the data frame with the list of medical record numbers that have Seizure Load values at baseline greater than x and x is the value you want Seizure Load to be greater than (e.g. 0).

source("https://raw.githubusercontent.com/borumlab/functions/master/baseline\_sz.R")  
baseline\_list\_seizures <- baseline\_sz(x)

# clean\_anthros.R

**Purpose**: Retains anthropometric values calculated from organizations as recommended by CDC, ASPEN, AND

**What you need to know**: This function uses the table called anthropometrics\_id\_research and the column headers "DAY\_TYPE", "AGE\_MO", "AGE\_DOL", WHO\_HT\_Z\_DAY", "CDC\_HT\_Z\_DAY", "NHANES\_HT\_Z\_DAY", "WHO\_WT\_Z\_DAY", "CDC\_WT\_Z\_DAY", "NHANES\_WT\_Z\_DAY", "WHO\_BMI\_Z\_DAY", "CDC\_BMI\_Z\_DAY", "NHANES\_BMI\_Z\_DAY", "WHO\_UAC\_Z\_DAY", "NHANES\_UAC\_Z\_DAY", "WHO\_TSF\_Z\_DAY", "NHANES\_TSF\_Z\_DAY", "WHO\_SSF\_Z\_DAY", "NHANES\_SSF\_Z\_DAY". This function will only keep height, weight and bmi z-score fro WHO for children <2 years of age and CDC fo children >= 2 years of age up to 20 years of age, and NHANES after 20 years of age. This function will only keep UAC, TSF, and SSF from WHO if <=1856 days of life otherwise it will use NHANES data.

**How to use**: Paste the lines below into your R code and modify per criteria where anthropometrics\_id\_research is the name of the data frame where you want the "cleaned" anthropometrics to go.

source("https://raw.githubusercontent.com/borumlab/functions/master/clean\_anthros.R")  
anthropometrics\_id\_research <- clean\_anthros()

# deidentify.R

**Purpose**: Deidentify patient medical record numbers with a KGID and deletes other identify columns with date

**What you need to know**: This function uses the table kgid\_mrnumber\_link\_research and columns "MRNUMBER"" and "KGID" to deidentify any table that has MRNUMBER. This function also deletes the following identifier colums: PKT\_INITIATED\_DATE, DOB,DATE,MRNUMBER. This function cannot be used for the demographics sheet since there are additional identifiers in that sheet that are not met with this function.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name\_deidentify is the name of the data frame where you want the deidentified data to go and table\_name is the name of the data frame you want to deidentify.

source("https://raw.githubusercontent.com/borumlab/functions/master/deidentify.R")  
table\_name\_deidentify <- deidentify(table\_name)

# deidentify\_demo.R

**Purpose**: Deidentify patient information in the demogrpahics table

**What you need to know**: This function uses the table demographics\_id\_calculated and columns that include date, name, city, state, zip code, county.

**How to use**: Paste the lines below into your R code and modify per criteria where demographics\_id\_calculated is the name of the data frame where you want the deidentified data to go.

source("https://raw.githubusercontent.com/borumlab/functions/master/deidentify\_demo.R")  
demographics\_id\_calculated <- deidentify\_demo()

# desc\_stat.R

**Purpose**: To calculate descriptive statistics on a parameter within a population

**What you need to know**: This function requires the name of the data frame and column and time frame you wish to do descriptive statistics. This function calculates the following descriptive statistics: total number of values, mean, median, standard deviation, standard error, maximum, minimum, range, 5th percetnile, 25th percentile, 50th percentile, 75th percentile, 95th percentile, variance, and coefficient of variation. It will only calculate descriptive stats on one time frame at a time.

**How to use**:Paste the lines below into your R code and modify per criteria where table name is the name of the data frame you want the descriptive statistics to go, df is the name of the data frame you want to do the descriptive statistics on, column name is the name of the column within the data frame you want to do the descriptive statistics on, and DOPKT\_value is a numerical value you want to do descriptive statistics on (e.g. 30) you may also enter "all" and this function will do descriptive statistics on all data with the specified data fram and column name.

source("https://raw.githubusercontent.com/borumlab/functions/master/desc\_stat.R")  
table\_name <- desc\_stat(df, "column name", DOPKT\_value)

# ifc.R

**Purpose**: Subsets a population that signed the informed consent before x date

**What you need to know**: This function uses the table demographics\_id\_calculated and column header called "PKT\_PROSPECTIVE\_DATE". This will eliminate all patients that did not sign the informed consent before x date from the table.

**How to use**: Paste the lines below into your R code and modify per criteria where demographics\_id\_calculated is the name of the data frame where you want the patients that meet the criteria to go and mm/dd/yyyy is the date you want all patients to have signed the informed consent by.

source("https://raw.githubusercontent.com/borumlab/functions/master/ifc.R")  
demographics\_id\_calculated <- ifc('mm/dd/yyyy')

# irb.R

**Purpose**: Subsets a population that is on a x IRB protocol number

**What you need to know**: This function uses the table demographics\_id\_calculated and column header called "IRB\_NUMBER". This will eliminate all patients that are not on x IRB protocol number.

**How to use**: Paste the lines below into your R code and modify per criteria where demographics\_id\_calculated is the name of the data frame where you want the patients that meet the criteria to go and TYPE IRB# HERE is the IRB protocol number you want patients to be on.

source("https://raw.githubusercontent.com/borumlab/functions/master/irb.R")  
demographics\_id\_calculated <- irb("TYPE IRB# HERE")

# medscore.R

**Purpose**: Calculate med score for time intervals equal to x number of days

**What you need to know**: This function uses the tables med\_data\_id\_research and a data frame with the list of medical record numbers you want to use after going through inclusion and exclusion criteria. If you enter 30 as the time period value then med score will be calculated for time intervals equal to 30 days where 1 is the beginning of the time period and 30 is the end of the time period. Refer to manuscript x for med score calculations.

**How to use**:Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclation to go and time period value is the numerical value you want med score calulcated at specified intervals, and population list is the name of the data frame with the listo f medical record numbers that meet the inclusion and exclusion criteria.

source("https://raw.githubusercontent.com/borumlab/functions/master/medscore.R")  
table\_name<-medscore(time period value, population list)

# mrlist.R

**Purpose**: Obtain list of medical record numbers from different data frames after applying inclusion and exclusion criteria.

**What you need to know**: This function uses up to 5 data frames, e.g. (demo\_list, baseline\_list\_seizure, baseline\_list\_meds, baseline\_list\_food, baseline\_list\_bmi) to get a collective list of medical record numbers after each data frame has undergone inclusion and exclusion criteria. It uses the MRNUMBER column in each data frame to create a list of medical record numbers.

**How to use**: Paste the lines below into your R code and modify per criteria where pop0000xx is the name of the data frame where you want the list of medical record numbers to go and a, b, c, d, e is the name of each data frame you want to create the list of medical record numbers from.

source("https://raw.githubusercontent.com/borumlab/functions/master/mrlist.R")  
pop0000xx <- mrlist(a,b,c,d,e)

# outcomescore.R

**Purpose**: Calculate outcome score from previously calculated seizure score and med score tables.

**What you need to know**: This function uses the tables seizure\_table\_deidentified and med\_table\_deidentified and column headers "KGID", "DOPKT", "SEIZURE\_SCORE", and "MED\_SCORE". See manuscript x for more details on the calculation. The result of this function will give a data frame with outcome score, seizure score, and med score for KGID and DOPKT.

**How to use**: Paste the lines below into your R code and modify per criteria where table name is the name of the data frame where you want the calculated outcome score to go and seizure table name and med table name are the names of the seizure table and med table you want to calculate outcome score from.

source("https://raw.githubusercontent.com/borumlab/functions/master/outcomescore.R")  
table\_name<-outcomescore(seizure table name, med table name)

# pkt\_bmi.R

**Purpose**: Select patients that have BMI available at desired time point on PKT (get rid of patients with missing data).

**What you need to know**: This function should only be used if the subset\_dopkt.R function has already been applied. It uses a modified anthropometrics\_id\_table. It will subset patient data if there is missing data based on the specified time point.

**How to use**: Paste the lines below into your R code and modify per criteria where pkt\_list\_bmi is the name of the data frame that lists MR numbers which meet your criteria and x months represents the amount of time on PKT that you want to be sure patients have data that is not missing.

source("https://raw.githubusercontent.com/borumlab/functions/master/pkt\_bmi.R")  
pkt\_list\_bmi <- pkt\_bmi(x months)

# query\_patient\_pkt.R

**Purpose**: To get tables from MySQL from a specified database and host where credentials are passed in.

**What you need to know**: Currently, this script is hard coded with hannah's username, database name patient\_pkt, and host IF-SRVV-BORUM. It is also hard coded to grab a list of table names as they are currently named in MySQL from the specified schema.

**How to use**: Paste the lines below into your R code. We have removed this from Github because we need to be able to pass database and credential information in without hardcoding it in. Currently, it is located in the directory listed below. Contacnt hannah for password information.

setwd("G:/MySQL Database/Bins/Queries/Functions/")  
source("query\_patient\_pkt.R")

# subset\_dopkt.R

**Purpose**: Subsets a population that was on PKT after x number of days.

**What you need to know**: This function uses the table demographics\_id\_calculated and column header called "MRNUMBER", "PKT\_INITIATED\_DATE", "DOB", and "DATE" in the data frame you pass into the function. This function will first convert date to day on PKT (create "DOPKT" column) and then remove all patient data where patients were not on PKT after x number of days.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name\_dopkt is the name of the data frame where you want the patients that meet the criteria to go and table\_name is the name of the dataframe you want to apply the criteria to and specify days on pkt is the number of days on PKT you want patients to have at least been on (it will remove all patients that are less than this number).

source("https://raw.githubusercontent.com/borumlab/functions/master/subset\_dopkt.R")  
table\_name\_dopkt <- subset\_dopkt(table\_name, specify days on pkt)

# subset\_pop.R

**Purpose**: Subsets a population for each data table based on list of medical record numbers that meet inclusion and exclusion criteria.

**What you need to know**: This function uses only column header called "MRNUMBER" in the data frames you pass in. This cannot be used unless subset\_dopkt is used before this function is used.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name\_pop is the name of the data frame where you want the patients that meet the criteria to go and table\_name is the name of the dataframe you want to apply the criteria to and pop0000xx is the name of the dataframe that has the list of medical record numbers that meet inclusion and exclusion criteria).

source("https://raw.githubusercontent.com/borumlab/functions/master/subset\_pop.R")  
table\_name\_pop <- subset\_pop(table\_name, pop0000xx)

# szscore.R

**Purpose**: Calculate seizure score for time intervals equal to x number of days

**What you need to know**: This function uses the tables seizure\_data\_id\_research and a data frame with the list of medical record numbers you want to use after going through inclusion and exclusion criteria. If you enter 30 as the time period value then seizure score will be calculated for time intervals equal to 30 days where 1 is the beginning of the time period and 30 is the end of the time period. Refer to manuscript x for seizure score calculations.

**How to use**:Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclation to go and time period value is the numerical value you want seizure score calulcated at specified intervals, and population list is the name of the data frame with the list of medical record numbers that meet the inclusion and exclusion criteria.

source("https://raw.githubusercontent.com/borumlab/functions/master/szscore.R")  
table\_name<-szscore(time period value, population list)

# total\_time\_on\_pkt.R

**Purpose**: Subsets a population that was on PKT with data before x date and up to x days on PKT

**What you need to know**: This function uses the tables demographics\_id\_calculated with column headers "PKT\_STOPPED\_DATE" and "PKT\_INITIATED\_DATE" and any data frame you pass into it. This function will create a column header called "TOTAL\_DAYS\_PKT". Medical record numbers will be eliminated if they were not on PKT before x date and have data up to x days on PKT.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclations to go and mm/dd/yyyy is the latest date to use to get x day s on PKT and specify number of days is the least number of days you wants patients to have been on PKT.

source("https://raw.githubusercontent.com/borumlab/functions/master/total\_time\_on\_pkt.R")  
table\_name <- total\_time\_on\_pkt('mm/dd/yyyy',specify number of days)

# response\_level.R

**Purpose**: Function will group patients on Seizure Score based on response level cutoffs for 4 categories

**What you need to know**: This function only uses the Seizure Score and you must put in 4 different categories to group patients. There must be a column called SEIZURE\_SCORE in your dataframe in order for this function to work. You must enter whole numbers for the response level category

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name is the data frame you want to modify, df is the name of the data frame with Seizure Score column and x is the lowest number in the response level category (e.g. 10), y is the second lowest number in the response level category (e.g. 50), z is the third lowest number in the response level category (e.g. 80), and a must be equal to z since a is the highest number in the response level category.

source("https://raw.githubusercontent.com/borumlab/functions/master/response\_level.R")  
table\_name <- response\_level(df,x,y,z,a)

# subset\_date.R

**Purpose**: Removes data after desired date on PKT

**What you need to know**: This function will remove all data after date entered into this function. This function may be useful to remove data based on a decided upon last date in the population. The data frame you will apply this to will likely be the demographics data frame. It needs to have a column header called DATE.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclations to go and x is the data frame you want to apply the calculations to (e.g. demographics data frame) and y is the last date you want data for the population and must be written as mm//dd/yyyy.

source("https://raw.githubusercontent.com/borumlab/functions/master/subset\_date.R")  
table\_name <- subset\_Date(x,y)

# demoDesc.R

**Purpose**: Calculates the frequency of a parameter in the demographics table

**What you need to know**: You must use the demographics data frame and column names. This is helpful to calculate the percent of patients for each demographic type within a population.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclations to go and df is the name of the data frame you want to apply the calculations to (e.g. demographics) and col="column" is the column name in the data frame you want to calculate the percent of a particular demographic on.

source("https://raw.githubusercontent.com/borumlab/functions/master/demoDesc.R")  
table\_name <- demoDesc(df, col="column")

# respStatus.R

**Purpose**: Assigns response levels based on Seizure, Med, or Outcome Score values. The Response levels it assigns are level 1: when score is 0-<=10%, level 2: when score is >10%-<=80%, level 3: when score is >80%

**What you need to know**: You must use a data frame with column headers called SEIZURE\_SCORE, MED\_SCORE\_ OUTCOME\_SCORE. Values in the columns cannot be blank in order for it to assign response levels. When it assigns a response level it will create a new column called SEIZURE\_RESPONSE, MED\_RESPONSE, and OUTCOME\_RESPONSE for SEIZURE\_SCORE, MED\_SCORE, and OUTCOME\_SCORE, respectively. It will then input the text "level1" "level2", or "level3" for response level 1, level 2, and level 3 respectively.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclations to go and df is the name of the data frame you want to apply the calculations to (e.g. demographics) and col="column" is the column name in the data frame you want to calculate the response level for (e.g. SEIZURE\_SCORE, MED\_SCORE, OUTCOME\_SCORE.

source("https://raw.githubusercontent.com/borumlab/functions/master/respStatus.R")  
table\_name <- respStatus(df, col = "column")

# respStatus2.R

**Purpose**: Assigns response levels within those assigned response level 2 from respStatus.R function based on Seizure, Med, or Outcome Score values. The Response levels it assigns are level 2\_1: when score is >10%-<=50% and level 2\_2: when score is >50%-<=80%

**What you need to know**: I apply the function respStatus to a data frame first and then subset the data frame to only include those values that were response level 2. I must then remove the SEIZURE\_RESPONSE, MED\_RESPONSE, and OUTCOME\_RESPONSE columns to run this function. Once that happens, then you must use a data frame with column headers called SEIZURE\_SCORE, MED\_SCORE\_ OUTCOME\_SCORE. Values in the columns cannot be blank in order for it to assign response levels. When it assigns a response level it will create a new column called SEIZURE\_RESPONSE, MED\_RESPONSE, and OUTCOME\_RESPONSE for SEIZURE\_SCORE, MED\_SCORE, and OUTCOME\_SCORE, respectively. It will then input the text "level2\_1" or "level2\_2" for response level 2\_1 or level 2\_2 respectively. You must then rbind this data frame with the data frames that have response level 1 and response level 3 if you want a data frame with all 4 response levels.

**How to use**: Paste the lines below into your R code and modify per criteria where table\_name is the name of the data frame where you want the calclations to go and df is the name of the data frame you want to apply the calculations to (e.g. demographics) and col="column" is the column name in the data frame you want to calculate the response level for (e.g. SEIZURE\_SCORE, MED\_SCORE, OUTCOME\_SCORE.

source("https://raw.githubusercontent.com/borumlab/functions/master/respStatus2.R")  
table\_name <- respStatus2(df, col = "column")