**Appendix – R-script**

In order to launch the software, you need to use R or R Studio and download it with the version compatible with your computer's operating system:

R: https://cloud.r-project.org or

R Studio: https://rstudio.com/products/rstudio.com/products/rstudio/download/

You will also need a database created in Access as the data loaded into the program is a .csv file

The column headers in the Access database used in this study are:

ID |study\_author\_yr |I\_literature |data\_source |study\_design |selection\_bias |Country |stroke\_type |disease\_definition |Study\_period\_start |Study\_period\_end |Study metadata\_comment |URL |gender\_ratio\_FM |age\_range |I\_0d\_28d |I\_29d\_1y |I\_1y\_4y |I\_5y\_9y |I\_10y\_17y |measurement |case\_counts |denominator |Rate |rate\_CI |unit |Study details\_comment |QC\_status |

################# Filter criteria #######################

get\_filter\_criteria = function(data){

 # Rate does not equal 0

 bool1 = data$Rate != 0

 # `measurement` contains the keyword "incidence" or "prevalence" (case-insensitive)

 bool2a = str\_detect(string = data$measurement, pattern = "[Ii]ncidence|[Pp]revalence")

 # Lo,2009 without 30 d - 20 y and 15 y - 20 y

 bool2b = str\_detect(string = data$study\_author\_yr, pattern = "Lo, 2009") &

 str\_detect(string = data$age\_range, pattern = "20 y", negate = TRUE)

 # OR operation

 bool2 = bool2a | bool2b

 # `stroke\_type` contains "AIS" and "Ischemic stroke"

 bool3 = str\_detect(string = data$stroke\_type, pattern = "AIS|[Ii]schemic stroke")

 # records that overlap multiple age groups

 bool4 = data$I\_0d\_28d + data$I\_29d\_1y + data$I\_1y\_4y + data$I\_5y\_9y + data$I\_10y\_17y > 1

 # numerator and denominator are both 0

 bool5 = (data$case\_counts == 0 & data$denominator == 0)

 # age group is specific to 0 d - 28 d

 bool6a = {data$I\_0d\_28d == TRUE} &

 {data$I\_29d\_1y + data$I\_1y\_4y + data$I\_5y\_9y + data$I\_10y\_17y == 0}

 # age range is between 0 d - 1 y and the `age\_group` column contains words like 29/30/31 days or 1 month

 bool6b = {data$I\_0d\_28d + data$I\_29d\_1y == 2} &

 {data$I\_1y\_4y + data$I\_5y\_9y + data$I\_10y\_17y == 0} &

 str\_detect(string = data$age\_range, pattern = "29|30|31|1 m")

 # OR evaluation

 bool6 = bool6a | bool6b

 # records that span all age groups (0 d - 18 y OR 29 d - 18 y)

 bool7 = data$I\_29d\_1y + data$I\_1y\_4y + data$I\_5y\_9y + data$I\_10y\_17y == 4

 return(list("bool1" = bool1, "bool2" = bool2, "bool3" = bool3, "bool4" = bool4, "bool5" = bool5, "bool6" = bool6, "bool7" = bool7))

}

describe\_data = function(data){

 # How many records are kept?

 print(paste0("Records: ", nrow(data)))

 # Percentage of literature?

 print(paste0("Percentage of literature: ", round(mean(data$I\_literature), digits = 4) \* 100, "%" ))

 # Data source

 print("These data sources are included:")

 print(table(data$data\_source))

 # How many records are there per country?

 print("These countries are included:")

 print(table(data$Country))

 # How many records are there per stroke type?

 print("These stroke types are included:")

 print(table(data$stroke\_type))

 # How many records are there per measurement?

 print("These measurments are included:")

 print(table(data$measurement))

 # How many records are there per unit?

 print("These units are included:")

 print(table(data$unit))

 # Check the study period

 sprintf(fmt = "Study period: %d - %d", min(data$Study\_period\_start), max(data$Study\_period\_end))

}

################# Country/Region ###############

# Determine each record's region (EU, UK, US, ROW)

get\_region = function(data){

 # Define EU members

 EU = c("Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Ireland, Italy, Malta, the Netherlands, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland") %>%

 str\_split(string = ., pattern = ",", simplify = TRUE) %>%

 str\_trim()

 # Determine whether a country is from EU, UK, US, or ROW

 bool\_US = data$Country == "United States"

 bool\_EU = data$Country %in% EU

 bool\_UK = data$Country == "United Kingdom"

 bool\_CAN = data$Country == "Canada"

 bool\_TWN = data$Country == "Taiwan"

 bool\_ROW = !bool\_US & !bool\_EU & !bool\_UK & !bool\_CAN & !bool\_TWN

 # Create a vector to indicate country type

 region = ""

 region[bool\_US] = "US"

 region[bool\_EU] = "Eur"

 region[bool\_UK] = "UK"

 region[bool\_CAN] = "Can"

 region[bool\_TWN] = "Twn"

 region[bool\_ROW] = "ROW"

 # Add the vector into the dataset as a column

 return(region)

}

# A function that summarizes the dataset by country or country type

summarize\_country = function(data, group){

 if(group == "all"){

 # Calculate mean IR, SEM, total number of cases, total number of studies for each country

 a = data %>%

 group\_by(Country) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr)))

 # # Order the factor variable 'Country'

 # a$Country = factor(x = a$Country, levels = fct\_inorder(a$Country))

 } else if (group == "region"){

 # Calculate mean IR, SEM, total number of cases, total number of studies for each country type

 a = data %>%

 group\_by(region) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr))) %>%

 rbind.data.frame(., tibble(region = "All regions",

 mean\_IR = mean(data$Rate),

 SEM = sd(data$Rate)/sqrt(nrow(data)),

 case\_total = sum(data$case\_counts),

 study\_total = length(unique(data$study\_author\_yr))))

 # Order the factor variable 'Country type'

 #a$region = factor(x = a$region, levels = c("EU","UK","US", "ROW", "All regions"))

 a$region = factor(x = a$region, levels = c("Can","Eur","Twn","UK","US","ROW","All regions"))

 } else if (group == "literature"){

 a = data %>%

 group\_by(I\_literature) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr))) %>%

 rbind.data.frame(., tibble(I\_literature = "All studies",

 mean\_IR = mean(data$Rate),

 SEM = sd(data$Rate)/sqrt(nrow(data)),

 case\_total = sum(data$case\_counts),

 study\_total = length(unique(data$study\_author\_yr))))

 a[a$I\_literature == "TRUE","I\_literature"] = "Literature"

 a[a$I\_literature == "FALSE","I\_literature"] = "Database"

 a$I\_literature = factor(a$I\_literature, levels = c("Database","Literature","All studies"))

 }

 else {

 stop("Undefined type. Choose 'all', 'region', or 'literature'.")

 }

 # Final touch: Replace 0 with NA in column 'case\_total'

 a$case\_total[a$case\_total == 0] = NA

 return(a)

}

################# Back-calculate numerators/denominators ####################

# Back-calculate any missing denominators and fill them in the dataset

fill\_denom = function(data){

 # Logical evaluation: which denominators are missing?

 bool\_denom = data$denominator == 0

 # For those missing denominators, do the back-calculation

 data$denominator[bool\_denom] = data$case\_counts[bool\_denom] / data$Rate[bool\_denom] \* 100000

 return(data)

}

# Back-calculate any missing numerators and fill them in the dataset

fill\_numer = function(data){

 # Logical evaluation: which numerators are missing?

 bool\_numer = data$case\_counts == 0

 # For those missing numerators, do the back-calculation

 data$case\_counts[bool\_numer] = round(data$Rate[bool\_numer] \* data$denominator[bool\_numer] / 100000)

 return(data)

}

################ Age ###################

# Create a function that converts binary indicators to character indicators

bi2char = function(data){

 # Check point: make sure records with multiple age groups are excluded

 if(sum(data$I\_0d\_28d + data$I\_29d\_1y + data$I\_1y\_4y + data$I\_5y\_9y + data$I\_10y\_17y) > nrow(data)){

 stop("There are records that cover multiple age groups.")

 }

 a = vector(mode = "character", length = nrow(data))

 a[data$I\_0d\_28d] = "0 d - 28 d"

 a[data$I\_29d\_1y] = "29 d - 1 y"

 a[data$I\_1y\_4y] = "1 y - 4 y"

 a[data$I\_5y\_9y] = "5 y - 9 y"

 a[data$I\_10y\_17y] = "10 y - 17 y"

 return(factor(x = a, levels = c("0 d - 28 d", "29 d - 1 y", "1 y - 4 y", "5 y - 9 y", "10 y - 17 y")))

}

# Which countries have data for all the age groups so that we can plot IR line curve for each country?

which\_country = function(data){

 # Summarize the data to list the number of observations at each age group for each country

 a = data %>%

 group\_by(Country) %>%

 summarize(n1 = sum(I\_0d\_28d), n2 = sum(I\_29d\_1y),

 n3 = sum(I\_1y\_4y), n4 = sum(I\_5y\_9y),

 n5 = sum(I\_10y\_17y))

 # Find countries where there's at least one age group that doesn't have any data

 bool = {a$n1 == 0} | {a$n2 == 0} | {a$n3 == 0} | {a$n4 == 0} | {a$n5 == 0}

 # Remove those countries

 return(a$Country[!bool])

}

summarize\_age = function(data, group){

 # Add a column for age indicator

 a = data %>%

 mutate(I\_age = bi2char(.))

 if(group == "all"){

 # Calculate mean IR, SEM, total case counts in each age group for all countries

 b = a %>%

 group\_by(I\_age) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr))) %>%

 dplyr::arrange(I\_age)

 } else if (group == "region"){

 # Calculate mean IR, SEM, total case counts in each age group for each region (US, EU, ROW)

 b = a %>%

 group\_by(region, I\_age) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr)))

 # Order the factor variable 'region'

 b$region = factor(x = b$region, levels = c("Can","Eur","Twn","UK","US","ROW"))

 } else if (group == "individual"){

 countries\_selected = which\_country(data)

 b = a %>%

 dplyr::filter(Country %in% countries\_selected) %>%

 group\_by(Country, I\_age) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr)))

 } else if (group == "literature"){

 a[a$I\_literature == "TRUE","I\_literature"] = "Literature"

 a[a$I\_literature == "FALSE","I\_literature"] = "Database"

 b = a %>%

 group\_by(I\_literature, I\_age) %>%

 summarise(mean\_IR = mean(Rate),

 SEM = sd(Rate)/sqrt(n()),

 case\_total = sum(case\_counts),

 study\_total = length(unique(study\_author\_yr)))

 } else {

 stop("Undefined group. Choose 'all', 'region', or 'literature'.")

 }

 # Final touch: Replace 0 with NA in column 'case\_total'

 b$case\_total[b$case\_total == 0] = NA

 return(b)

}

################ Confidence Interval ##############################

# Create a function to calculate Poisson CI

calc\_Pois\_CI = function(numerator, denominator, alpha){

 IR\_lb = qchisq(p = alpha/2, df = 2 \* numerator) \* 100000 / (2 \* denominator)

 IR\_ub = qchisq(p = 1 - alpha/2, df = 2 \* (numerator + 1)) \* 100000 / (2 \* denominator)

 return(list(lb = IR\_lb, ub = IR\_ub))

}

# Create a helper function to calculate the really complex ratio for Wilson CI

f\_wils\_ratio = function(p\_hat, n, z){

 return((p\_hat + z ^ 2 / (2\*n) + z \* sqrt((p\_hat \* (1-p\_hat))/n + z^2/(4\*n^2))) / (1 + z ^ 2 / n))

}

# Create a function to calculate Wilson CI

calc\_Wils\_CI = function(numerator, denominator, alpha){

 p\_hat = numerator / denominator

 z\_lb = qnorm(p = alpha/2, mean = 0, sd = 1, lower.tail = TRUE)

 z\_ub = qnorm(p = 1 - alpha/2, mean = 0, sd = 1, lower.tail = TRUE)

 IR\_lb = f\_wils\_ratio(p\_hat = p\_hat, n = denominator, z = z\_lb) \* 100000

 IR\_ub = f\_wils\_ratio(p\_hat = p\_hat, n = denominator, z = z\_ub) \* 100000

 return(list(lb = IR\_lb, ub = IR\_ub))

}

# Create a function to format the CI

format\_CI = function(data, digits = 2){

 return(str\_c(format(x = round(x = data$lb, digits = digits), nsmall = 2), format(x = round(x = data$ub, digits = digits), nsmall = 2), sep = " - "))

}

# Wrapper function

get\_CI = function(data, type, alpha = 0.05, digits = 2){

 # Calculate CI

 if(type == "Poisson"){

 a = calc\_Pois\_CI(numerator = data$case\_counts, denominator = data$denominator, alpha = alpha)

 } else if (type == "Wilson"){

 a = calc\_Wils\_CI(numerator = data$case\_counts, denominator = data$denominator, alpha = alpha)

 } else {

 stop("Undefined CI type. Choose 'Poisson' or 'Wilson'.")

 }

 # Format CI

 return(format\_CI(data = a, digits = digits))

}

# Fill any missing CIs

fill\_CI = function(data, type, alpha, digits){

 # Logical evaluation: which CIs are missing?

 bool\_CI = data$rate\_CI == ""

 data$rate\_CI[bool\_CI] = get\_CI(data = data[bool\_CI,], type = type, alpha = alpha, digits = digits)

 return(data)

}

################# Data Visualization ####################

# Bar chart: visualize IR grouped by a factor variable

vis\_bar = function(data, var, alpha, xlab, ylab, CI = FALSE,

 angle = NULL, vjust = NULL, hjust = NULL,

 size = NA, nudge\_y = NULL, nudge\_x = NULL){

 # Order the bars

 if(var == "Country"){

 temp = data %>%

 dplyr::arrange(desc(mean\_IR))

 temp$Country = fct\_inorder(f = as.factor(temp$Country))

 tcklabels = paste0(as.character(temp[,var]$Country),"\n","N = ",temp$study\_total)

 } else if (var == "region"){

 temp = data

 temp$region = droplevels(temp$region)

 tcklabels = paste0(as.character(temp[,var]$region),"\n","N = ",temp$study\_total)

 } else if (var == "I\_literature"){

 temp = data

 tcklabels = paste0(as.character(temp[,var]$I\_literature),"\n","N = ",temp$study\_total)

 } else {

 stop("Undefined variable. Choose 'region', 'Country', or 'I\_literature'.")

 }

 plot = ggplot(data = temp, aes\_string(x = var)) +

 geom\_col(aes(y = mean\_IR)) +

 # geom\_text(aes(y = mean\_IR, label = paste0("n = ", case\_total)),

 # nudge\_y = nudge\_y, nudge\_x = nudge\_x, size = size) +

 geom\_text(aes(y = mean\_IR, label = round(mean\_IR, digits = 2)),

 nudge\_y = nudge\_y, nudge\_x = nudge\_x, size = size) +

 labs(x = xlab, y = ylab) +

 theme\_bw() +

 theme(axis.text.x = element\_text(angle = angle, vjust = vjust,

 hjust = hjust, face = "bold", size = 12),

 legend.position = "top") +

 scale\_x\_discrete(labels=tcklabels)

 if (CI == TRUE){

 plot = plot +

 geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15)

 }

 return(plot)

}

### Line chart: visualize IR for different age groups

# If var == "all", then visualize IR for all countries

# If var == "region", then visualize IR for each country type (e.g. EU, US, ROW)

vis\_line = function(data, var, alpha, xlab, ylab, nudge\_y = NULL, nudge\_x = NULL, style = "color", manual = FALSE, values){

 if(var == "all"){

 ggplot(data = data, aes(x = I\_age, y = mean\_IR)) +

 geom\_line(aes(group = 1)) +

 geom\_point() +

 geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 #geom\_text(aes(label = study\_total), nudge\_y = nudge\_y, nudge\_x = nudge\_x) +

 labs(x = xlab, y = ylab) +

 theme\_bw() +

 theme(legend.position = "top", axis.text.x = element\_text(face = "bold", size = 12))

 } else if (var == "region"){

 if(style == "color"){

 #nudge\_factor = as.numeric(data$region)

 #nudge\_factor[nudge\_factor == 2] = 1

 #nudge\_factor[nudge\_factor == 4] = 0

 #nudge\_factor[nudge\_factor == 5] = -1

 ggplot(data = data, aes(x = I\_age, y = mean\_IR, color = region)) +

 geom\_line(aes(group = region)) +

 geom\_point() +

 #geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 # ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 #geom\_text(aes(label = study\_total), nudge\_y = nudge\_y\*nudge\_factor, nudge\_x = nudge\_x, show.legend= FALSE) +

 labs(x = xlab, y = ylab, color = "Region") +

 theme\_bw() +

 theme(legend.position = "top",

 axis.text.x = element\_text(face = "bold", size = 12),

 legend.text = element\_text(face = "bold", size = 12))

 } else if (style == "linetype"){

 # Option for specifying line types

 if(manual == FALSE){

 ggplot(data = data, aes(x = I\_age, y = mean\_IR, group = region)) +

 geom\_line(aes(linetype = region)) +

 geom\_point() +

 #geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 # ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 labs(x = xlab, y = ylab, linetype = "Region") +

 # scale\_linetype\_manual(values=c("twodash", "dotted"))+

 theme\_bw() +

 theme(legend.position = "top",

 axis.text.x = element\_text(face = "bold", size = 12),

 legend.text = element\_text(face = "bold", size = 12))

 } else {

 my\_linetype = values

 ggplot(data = data, aes(x = I\_age, y = mean\_IR, group = region)) +

 geom\_line(aes(linetype = region)) +

 geom\_point() +

 #geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 # ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 labs(x = xlab, y = ylab, linetype = "Region") +

 scale\_linetype\_manual(values = my\_linetype)+

 theme\_bw() +

 theme(legend.position = "top",

 axis.text.x = element\_text(face = "bold", size = 12),

 legend.text = element\_text(face = "bold", size = 12))

 }

 } else if (style == "shape"){

 ggplot(data = data, aes(x = I\_age, y = mean\_IR, group = region)) +

 geom\_line() +

 geom\_point(aes(shape = region)) +

 #geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 # ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 # geom\_text(aes(label = case\_total), nudge\_y = nudge\_y, nudge\_x = nudge\_x) +

 labs(x = xlab, y = ylab, shape = "Region") +

 theme\_bw() +

 theme(legend.position = "top",

 axis.text.x = element\_text(face = "bold", size = 12),

 legend.text = element\_text(face = "bold", size = 12))

 } else {

 stop("Unknown style. Please choose 'color', 'linetype', or 'shape'")

 }

 } else if (var == "individual"){

 ggplot(data = data, aes(x = I\_age, y = mean\_IR, color = Country)) +

 geom\_line(aes(group = Country)) +

 geom\_point() +

 #geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 # ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 #geom\_text(aes(label = study\_total), nudge\_y = nudge\_y, nudge\_x = nudge\_x) +

 labs(x = xlab, y = ylab, color = "Country") +

 theme\_bw() +

 theme(legend.position = "top",

 axis.text.x = element\_text(face = "bold", size = 12),

 legend.text = element\_text(face = "bold", size = 12))

 } else if (var == "I\_literature"){

 #nudge\_factor = data$I\_literature

 #nudge\_factor[nudge\_factor == "Database"] = 1

 #nudge\_factor[nudge\_factor == "Literature"] = -1

 #nudge\_factor = as.numeric(nudge\_factor)

 ggplot(data = data, aes(x = I\_age, y = mean\_IR, color = I\_literature)) +

 geom\_line(aes(group = I\_literature)) +

 geom\_point() +

 geom\_errorbar(aes(ymin = mean\_IR - qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM,

 ymax = mean\_IR + qnorm(p = 1 - alpha/2, mean = 0, sd = 1) \* SEM), width = 0.15) +

 #geom\_text(aes(label = study\_total), nudge\_y = nudge\_y\*nudge\_factor, nudge\_x = nudge\_x, show.legend= FALSE) +

 labs(x = xlab, y = ylab, color = "Data Source") +

 theme\_bw() +

 theme(legend.position = "top",

 axis.text.x = element\_text(face = "bold", size = 12),

 legend.text = element\_text(face = "bold", size = 12))

 } else {

 stop("Undefined grouping variable. Choose 'all', 'region', or 'I\_literature'.")

 }

}

################### Meta-analysis #####################

# Calculate the standard errors (based on Neyeloff 2012 paper)

calc\_se = function(data){

 return((data$Rate/100000)/sqrt((data$Rate/100000) \* data$denominator) \* 100000)

}

# A function that calculates I^2 (based on Neyeloff 2012 paper)

calc\_I2 = function(data){

 if(nrow(data) == 1){

 return(NaN)

 } else{

 # Calculate the standard error

 se = calc\_se(data = data)

 # Weight

 w = 1/se^2

 # Q

 Q = t(w) %\*% data$Rate^2 - (t(w) %\*% data$Rate)^2 / (sum(w))

 #Q = t(w) %\*% (data$Rate - (t(w) %\*% data$Rate)/sum(w))^2

 # I^2

 deg\_f = nrow(data) - 1

 Isq = (Q - deg\_f) / Q \* 100

 return(max(as.vector(Isq), 0))

 }

}

# Get record's publication year

get\_pub\_year = function(data){

 a = str\_split(string = data$study\_author\_yr, pattern = ",", simplify = TRUE)[,2] %>%

 str\_squish() %>%

 str\_sub(string = ., start = 1, end = 4) %>%

 as.numeric()

 return(a)

}

# Summarize the dataset to prepare for forest plot visualization

summarize\_forest = function(data, type, alpha = 0.05, digits = 2){

 # Fill any missing numerators

 a = fill\_numer(data)

 # Fill any missing denominators

 b = fill\_denom(a)

 # Fill any missing CIs

 c = fill\_CI(data = b, type = type, alpha = alpha, digits = digits)

 return(c)

}

# Visualize the forest plot

my\_viz\_forest = function(data, type = "standard", variant = "classic", group,

 method = "FE", confidence\_level = 0.95,

 study\_labels = NA, summary\_label = "Summary effect", xlab,...){

 # Calculate the standard error

 se = calc\_se(data = data)

 # Create a temporary dataset

 a = data.frame(rate = data$Rate, se = se, study\_author\_yr = data$study\_author\_yr, literature = data$I\_literature,

 region = data$region, n = data$case\_counts)

 a$region = factor(x = a$region, levels = c("Can","Eur","Twn","UK","US","ROW"))

 # Determine grouping factor

 if(group == "all"){

 my\_group = NULL

 # Sort by publication year

 b = a %>%

 mutate(pub\_year = get\_pub\_year(data = a)) %>%

 dplyr::arrange(pub\_year)

 # Extract labels

 my\_label = paste0(as.character(b$study\_author\_yr), " (n = ", b$n, ")")

 # Remove extra columns

 temp = b %>%

 dplyr::select(c(rate, se))

 } else if (group == "region"){

 # Sort by publication year

 b = a %>%

 mutate(pub\_year = get\_pub\_year(data = a)) %>%

 group\_by(region) %>%

 dplyr::arrange(region, pub\_year)

 # Extract labels

 my\_label = paste0(as.character(b$study\_author\_yr), " (n = ", b$n, ")")

 my\_group = as.factor(b$region)

 # Remove extra columns

 temp = b %>%

 ungroup() %>%

 dplyr::select(c(rate, se))

 } else if (group == "I\_literature"){

 # Sort by publication year

 b = a %>%

 mutate(pub\_year = get\_pub\_year(data = a)) %>%

 group\_by(literature) %>%

 dplyr::arrange(literature, pub\_year)

 # Extract labels

 my\_label = paste0(as.character(b$study\_author\_yr), " (n = ", b$n, ")")

 my\_group = as.factor(b$literature)

 # Remove extra columns

 temp = b %>%

 ungroup() %>%

 dplyr::select(c(rate, se))

 }

 else {

 stop("Undefined group. Choose 'all' or 'region'.")

 }

 # If study\_labels == NA, then we use study\_author\_yr as the labels by default

 # Otherwise, we allow the user to provide a custom label vector

 if(is.na(study\_labels)){

 viz\_forest(x = as.data.frame(temp), type = type,

 variant = variant, method = method, confidence\_level = confidence\_level,

 group = my\_group, study\_labels = my\_label,

 summary\_label = summary\_label, xlab = xlab, text\_size = 3, col = "Greys", ...)

 } else {

 viz\_forest(x = temp, type = type,

 variant = variant, method = method, confidence\_level = confidence\_level, group = my\_group,

 study\_labels = study\_labels,

 summary\_label = summary\_label, xlab = xlab, text\_size = 3, col = "Greys",...)

 }

}

################ Tables ##################################

# Get records that are excluded from analysis

get\_excluded = function(before, after){

 subset(x = before[before$ID %in% setdiff(x = before$ID, y = after$ID),],

 select = c("study\_author\_yr", "age\_range","stroke\_type",

 "measurement", "case\_counts", "denominator",

 "Rate")) %>%

 dplyr::arrange(study\_author\_yr)

}

# Display a subset table with essential information

format\_table = function(data){

 # Format the table

 a = data %>%

 mutate(Study\_period = str\_c(Study\_period\_start, Study\_period\_end, sep = " - "),

 rate\_and\_CI = str\_c(round(Rate, digits = 2), " (",rate\_CI,")"),

 design\_and\_bias = str\_c(study\_design, " (", selection\_bias, ") ")) %>%

 dplyr::select(study\_author\_yr, region, Country, data\_source, design\_and\_bias, Study\_period, age\_range, case\_counts, rate\_and\_CI) %>%

 dplyr::arrange(region, Country)

 # Remove () in 'rate\_and\_CI' and 'design\_and\_bias'

 a$rate\_and\_CI = str\_remove(string = a$rate\_and\_CI, pattern = "\\(\\)")

 a$design\_and\_bias = str\_remove(string = a$design\_and\_bias, pattern = "\\(\\)")

 # Replace all the 0s in case\_counts with NA

 a$case\_counts[a$case\_counts == 0] = NA

 # Format the column names

 colnames(a) = c("Author, year", "Region","Country", "Data source","Study design (selection bias)", "Study period", "Age range", "Numberator (n)", "Incidence rate per 100k (95% CI)")

 return(a)

}

# Output the table using kable\_styling()

output\_table = function(data, format = "pandoc", title){

 kable\_styling(kable\_input = kable(x = data, format = format, caption = title), full\_width = TRUE)

}

############### Miscellaneous ################

explain\_cols = function(){

 colnames\_necessary = c("study\_author\_yr","I\_literature","data\_source","study\_design",

 "selection\_bias","Country","stroke\_type","Study\_period\_start",

 "Study\_period\_end","age\_range","I\_0d\_28d",'I\_29d\_1y','I\_1y\_4y',

 'I\_5y\_9y','I\_10y\_17y','measurement','case\_counts','denominator',

 'Rate','rate\_CI','unit')

 definition = c("Study's author and publication year",

 "Binary indicator: Is the record a literature study?",

 "Data source",

 "Study design",

 "Selection bias",

 "Country",

 "Stroke type (ischemic, hemorrhagic, etc.)",

 "Study period (start)",

 "Study period (end)",

 "Age range",

 "Binary indicator: Does the record cover 0 day - 28 days?",

 "Binary indicator: Does the record cover 29 days - < 1 year?",

 "Binary indicator: Does the record cover 1 year - 4 years?",

 "Binary indicator: Does the record cover 5 years - 9 years?",

 "Binary indicator: Does the record cover 10 years - 17 years?",

 "Measurement (incidence, prevalence, etc.)",

 "Numerator (number of cases)",

 "Denominator (population of the age subgroup)",

 "Incidence rate",

 "Confidence interval of the incidence rate",

 "Measurement unit (per 100k, per 100k live births, etc.)"

 )

 return(data.frame(colnames\_necessary, definition))

}

# Check if the dataset has all the necessary columns

get\_qc = function(data, logic = FALSE){

 # Get the necessary column names

 colnames\_necessary = explain\_cols()[["colnames\_necessary"]]

 # Find out whether the uploaded CSV has all the necessary columns

 bool = colnames\_necessary %in% colnames(data)

 # Output TRUE/FALSE (used for output$quality\_ui)

 if(logic == TRUE){

 return(all(bool) == TRUE)

 } else {

 if(all(bool) == TRUE){

 c("The imported dataset has PASSED the quality check.")

 } else {

 paste0("WARNING! The following columns are missing: ", str\_c(colnames\_necessary[!bool], collapse = ", "), ". Please read the introduction page for more information.")

 }

 }

}