```
# Forest Biomass in Bangladesh: An historical review of forest inventories to assist national estima

# Gael Sola, FAO, gael.sola@fao.org,
# Liam Costello, liam.costello@fao.org
# 04/2016

# Confidence interval for the biomass estimates

######################################################################
# Effacer memoire
rm(list=ls())

# Verification du working directory
getwd()

# remove graphs
dev.off()

# Load libraries
library(tidyverse)

# Setting working directory
setwd("/media/gs/OSDisk/Users/solag/Documents/Work/Bangladesh article harmo/Analysis/data")

# Set the plot theme
source("R-corrections/theme_gs_print.R")

#-----------------------------------------------------------------#
#               Read tables and checks
#-----------------------------------------------------------------#

# Read tables
plot_data_alltrees <- read.csv("results/plot_agb_alltrees.csv", header = T, stringsAsFactors = F)
plot_data_sup10    <- read.csv("results/plot_agb_sup10.csv", header = T, stringsAsFactors = F)
lccs_codes         <- read.csv("R-corrections/lccs_codes.csv", header = T, stringsAsFactors = F)


# Create an id that mixes lccs6 and entity_ab
plot_data_alltrees$mix <- paste(plot_data_alltrees$lccs6,
                                plot_data_alltrees$entity_ab,
                                sep=" --- ")
plot_data_sup10$mix    <- paste(plot_data_sup10$lccs6,
                                plot_data_sup10$entity_ab,
                                sep=" --- ")

#-----------------------------------------------------------------#
#                 Add Non-forested plots from NFA
#-----------------------------------------------------------------#

# Read table
nfa_no_tree <- read.csv("R-corrections/plots_without_tree_nfa.csv", header = T, stringsAsFactors = F

# Convert plot size to ha
nfa_no_tree$plot_size_ha <- nfa_no_tree$plot_size_ha/10000
summary(nfa_no_tree$plot_size_ha)

# Removing non vegetated plots
nfa_no_tree <- nfa_no_tree[-which(nfa_no_tree$lccs1 == "A"),]

# Removing no lccs6 plots (mostly annual crop)
nfa_no_tree <- nfa_no_tree[!is.na(nfa_no_tree$lccs6),]

# Check and adjust the columns to plot_data
names(plot_data)
names(nfa_no_tree)

nfa_no_tree$lus      <- NULL
nfa_no_tree$entity   <- NULL
nfa_no_tree$x_corr   <- NULL
```

```
nfa_no_tree$y_corr   <- NULL
nfa_no_tree$nstem_ha <- 0


#-----------------------------------------------------------------------------#
#     Create function to calculate weighted agb and confidence intervals
#-----------------------------------------------------------------------------#

calc_agb_ci <- function(mydata, nfa_no_tree = nfa_no_tree, myclass, remove_plots = T){

  plot_data    <- mydata
  nfa_no_tree  <- nfa_no_tree
  myclass      <- myclass
  remove_plots <- remove_plots

  # #temp
  # plot_data    <- plot_data_alltrees
  # nfa_no_tree  <- nfa_no_tree
  # myclass      <- "mix"
  # remove_plots <- T


  ###
  ### STEP 1: add NFA plots without trees to make non forest classes estimates more realistic
  ###
  # Create a vector of NFA plot id in the plot_data table
  plot_id <- plot_data[which(plot_data$entity_ab == "FD-07-NFA"), "id"]

  # remove these plots from
  nfa_no_tree2 <- nfa_no_tree[-which(nfa_no_tree$id %in% plot_id),]

  # Check the lccs classes contribution plots with no agb
  print("nb plots with no trees")
  print(summary(as.factor(nfa_no_tree2$lccs6)))

  ###
  ### STEP 2: Prepare the data
  ###
  # Remove plots for which project ad lccs class provide unreliable results
  plot_data2 <- plot_data

  if (remove_plots == T) {

    plot_data2 <- plot_data2[which(plot_data2$mix != "NMF --- FRMP-97")  ,]
    plot_data2 <- plot_data2[which(plot_data2$mix != "NMF --- FD-07-NFA"),]
    plot_data2 <- plot_data2[which(plot_data2$mix != "FP --- FD-07-NFA") ,]

  }

  # Initiate class
  plot_data2$class <- eval(parse(text = paste("plot_data2$",myclass , sep="")))

  # Remove all plots with class is NA
  plot_class <- plot_data2[which(!is.na(plot_data2$class)),c("id","class","agb_t_ha","plot_size_ha",

  ###
  ### STEP 3: calculate the weighted and flat agb average per class
  ###
  # TABLE: agb and area covered per class with weigthed average
  plot_class$plot_count <- 1
  plot_class$agbxarea <- plot_class$agb_t_ha*plot_class$plot_size_ha
  weighted_agb <- Reduce(function(...)merge(..., by = "class"),
                         list(a = aggregate(plot_count~class, data = plot_class, FUN= sum),
                              b = aggregate(plot_size_ha~class, data = plot_class, FUN= sum),
                              c = aggregate(agbxarea~class, data = plot_class, FUN= sum)
                              )
                         )

  names(weighted_agb)
```

```
names(weighted_agb)[3] <- "sum_area_plots"
names(weighted_agb)[4] <- "sum_agbxarea"

weighted_agb$agb_class_w <- weighted_agb$sum_agbxarea/weighted_agb$sum_area_plots

###
### STEP 4 : calculate ci95 (ratio estimator)
###
# Get back class weighted average agb to plot data and calculate (agbplot - agbclass)^2*areaplot^2
plot_class <- merge(plot_class, weighted_agb[,c("class", "agb_class_w")], by="class")
plot_class$sumsq <- (plot_class$agb_t_ha - plot_class$agb_class_w)^2*plot_class$plot_size_ha^2

# TABLE: calculate sumsq and average plot size per class
class_res <- Reduce(function(...)merge(..., by = "class"),
                    list(weighted_agb[,c("class", "plot_count", "sum_area_plots", "agb_class_w")],
                         a = aggregate(sumsq~class, data=plot_class, FUN=sum),
                         b = aggregate(plot_size_ha~class, data= plot_class, FUN=mean)
                         )
                    )
names(class_res)[6] <- "avg_area_plots"

# calculate variance standard error and conf int (ratio estimator)
class_res$var_w <- 1/(class_res$plot_count*(class_res$plot_count-1)*class_res$avg_area_plots^2)*cl
class_res$se_w <- sqrt(class_res$var)
class_res$ci95_mean_w <- qt(0.975, df=class_res$plot_count-1)*class_res$se

###
### STEP 5: Add flat agb and stdev
###
standard_agb <- Reduce(function(...)merge(..., by = "class"),
                       list(a = aggregate(agb_t_ha~class, data=plot_class, FUN=mean),
                            b = aggregate(agb_t_ha~class, data=plot_class, FUN=sd)
                            )
                       )

names(standard_agb)[2] <- "agb_class_flat"
names(standard_agb)[3] <- "agb_class_flatsd"

###
### STEP 6: Merge tables and prepare graph
###
# Merge tables
if(myclass == "mix") {

  class_res_final <- Reduce(function(...)merge(..., by = "class"),
                            list(a = unique(plot_data2[,c("class", "lccs6", "entity_ab")]), #activ
                                 b = aggregate(nstem_ha~class, data = plot_class, FUN=mean),
                                 c = aggregate(ba_ha~class, data = plot_class, FUN=mean),
                                 class_res[,!(names(class_res) %in% c("sumsq","avg_area_plots","va
                                 standard_agb
                                 )
                            )

  # Calculate CI for flat average
  class_res_final$agb_class_flatci95 <- qt(0.975, df=class_res_final$plot_count-1)*class_res_final

  # Remove classes with less than 2 plots
  class_res_final2 <- class_res_final[class_res_final$plot_count > 2,]

  # Plot
  res_plot <- ggplot(class_res_final2, aes(x=entity_ab, y=agb_class_w)) +
    geom_col() +
    geom_errorbar(aes(ymin=agb_class_w-ci95_mean_w, ymax=agb_class_w+ci95_mean_w), width=0.2) +
    geom_text(aes(y=agb_class_w+8,label=plot_count), position = position_dodge(0.9), vjust = 0, an
    xlab("Land cover and project classes") + ylab("Average biomass (t/ha)") +
    facet_grid(~lccs6, scales = "free_x", space = "free_x") + #activate only if class == mix
    theme(axis.text.x = element_text(angle = 90),
          axis.text.y = element_text(angle = 90),
          axis.title.x = element_text(angle = 180))
```

```
    res_table <- class_res_final2

  } else {

    class_res_final <- Reduce(function(...)merge(..., by = "class"),
                              list(b = aggregate(nstem_ha~class, data = plot_class, FUN=mean),
                                   c = aggregate(ba_ha~class, data = plot_class, FUN=mean),
                                   class_res[,!(names(class_res) %in% c("sumsq","avg_area_plots","va
                                   standard_agb
                              )
    )

    # Calculate CI for flat average
    class_res_final$agb_class_flatci95 <- qt(0.975, df=class_res_final$plot_count-1)*class_res_final

    # Remove classes with less than 2 plots
    class_res_final2 <- class_res_final[class_res_final$plot_count > 2,]

    # Plot
    res_plot <- ggplot(class_res_final2, aes(x=class, y=agb_class_w)) +
      geom_col() +
      geom_errorbar(aes(ymin=agb_class_w-ci95_mean_w, ymax=agb_class_w+ci95_mean_w), width=0.2) +
      geom_text(aes(y=agb_class_w+8,label=plot_count), position = position_dodge(0.9), vjust = 0, ar
      xlab("Land cover and project classes") + ylab("Average biomass (t/ha)") +
      theme(axis.text.x = element_text(angle = 90),
            axis.text.y = element_text(angle = 90),
            axis.title.x = element_text(angle = 180))


    res_table <- class_res_final2

  } # END OF IF

  res <- list(res_plot, res_table)
  return(res)

} # END OF THE FUNCTION



#-------------------------------------------------------------------------#
#       Apply the function to the different configurations tested
#-------------------------------------------------------------------------#

###
### all trees all plots, class = mix
###
calc_ci <- calc_agb_ci(mydata = plot_data_alltrees,
                       myclass = "mix", # column name entered as  text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = FALSE)

calc_ci[1]
# Save graph
ggsave("results/results_final/gr_agb_lccs6xentity_alltrees_allplots.png", width = 15, height = 9, un

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_lccs6xentity_alltrees_allplots.csv", row.names

###
### trees with dbh >= 10, all plots, class = mix
###
calc_ci <- calc_agb_ci(mydata = plot_data_sup10,
                       myclass = "mix", # column name entered as  text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = FALSE)
```

```
calc_ci[1]
# Save graph
ggsave("results/results_final/gr_agb_lccs6xentity_sup10_allplots.png", width = 15, height = 9, units

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_lccs6xentity_sup10_allplots.csv", row.names =

###
### all trees, all plots, class = lccs6
###
calc_ci <- calc_agb_ci(mydata = plot_data_alltrees,
                       myclass = "lccs6", # column name entered as text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = FALSE)

calc_ci[1]
# Save graph
#ggsave("results/results_final/gr_agb per lccs6_sup10_allplots.png", width = 15, height = 9, units =

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_lccs6_alltress_allplots.csv", row.names = F)

###
### all trees, removing plots in NMF and FP , class = lccs6
###
calc_ci <- calc_agb_ci(mydata = plot_data_alltrees,
                       myclass = "lccs6", # column name entered as text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = TRUE)

calc_ci[1]
# Save graph
ggsave("results/results_final/gr_agb_per_lccs6_alltrees_selectNMFandFP_FINAL.csv.png", width = 15, h

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_lccs6_alltrees_selectNMFandFP_FINAL.csv", row

###
### sup 10, removing plots in NMF and FP , class = lccs6
###
calc_ci <- calc_agb_ci(mydata = plot_data_sup10,
                       myclass = "lccs6", # column name entered as text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = TRUE)

calc_ci[1]
# Save graph
#ggsave("results/results_final/gr_agb_per_lccs6_sup10_selectNMFandFP.csv.png", width = 15, height =

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_lccs6_sup10_selectNMFandFP.csv", row.names = F


###
### all trees, removing plots in NMF and FP , class = bfi
###
calc_ci <- calc_agb_ci(mydata = plot_data_alltrees,
                       myclass = "bfi_zone", # column name entered as text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = TRUE)

calc_ci[1]
# Save graph
#ggsave("results/results_final/gr_agb_per_bfi_alltrees_selectNMFandFP_FINAL.csv.png", width = 15, he

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_bfi_alltrees_selectNMFandFP_FINAL.csv", row.na

###
```

```
### all trees, removing plots in NMF and FP , class = fao
###
calc_ci <- calc_agb_ci(mydata = plot_data_alltrees,
                       myclass = "fao_biome", # column name entered as text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = TRUE)

calc_ci[1]
# Save graph
#ggsave("results/results_final/gr_agb_per_fao_alltrees_selectNMFandFP_FINAL.csv.png", width = 15, he

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_fao_alltrees_selectNMFandFP_FINAL.csv", row.na

###
### NFA trees, all plots , class = lccs6
###
calc_ci <- calc_agb_ci(mydata = plot_data_alltrees[which(plot_data_alltrees$entity_ab == "FD-07-NFA"
                       myclass = "lccs6", # column name entered as text required
                       nfa_no_tree = nfa_no_tree,
                       remove_plots = TRUE)

calc_ci[1]
# Save graph
ggsave("results/results_final/gr_agb_per_lccs6_alltreesNFA_allplots_FINAL.csv.png", width = 15, heig

# Save plots
write.csv(calc_ci[2], file="results/results_final/agb_lccs6_alltreesNFA_allplots_FINAL.csv", row.nam
```