**A Brief Guide of the R code for Graphics in Chao et al. (2023) Paper**

This guide demonstrates how to make graphics shown in Figures 2 to 4 in Chao et al.’s (2023) Ecology Letters paper; all data and R code are available in Anne Chao’s Github repository at <https://github.com/AnneChao/ELE_MF.beta4>. To run the code, it is required R version > 4.0.0. Before using the data and code, the following packages on CRAN must be installed and imported:

library(lme4)

library(ggpubr)

library(lmerTest)

library(magrittr)

library(parallel)

library(reshape2)

library(tidyverse)

library(missForest)

library(RColorBrewer)

The source code for graphics (Figures 2 to 4) in Chao et al. (2023) paper is provided in Anne Chao’s Github repository. First, import/load the source code:

source("Source R code.txt")

1. **Figure 2: The patterns of within-plot multifunctionality measure of order q = 0, 1, 2 with respect to tree species diversity**

Figure 2 plots the patterns of within-plot multifunctionality measure of order q = 0 (left panels), q = 1 (middle panels), and q = 2 (right panels) with respect to tree species diversity of the same order q based on 26 ecosystem functions collected in a total of 209 plots in six European countries when all functions are assumed to be independent (upper panels in Figure 2), or when the correlation between any two functions is accounted for (lower panels in Figure 2). First, run the following code to read ecosystem functions data, and basal area data (as a proxy of species abundance to compute species diversity).

Forest\_function\_data\_raw = read.csv("Forest\_function\_data\_raw.csv")

Forest\_biodiversity\_data = read.csv("Forest\_biodiversity\_data.csv", sep = '"')[,c('block', 'plot', 'full\_species\_original', 'X.6')]

colnames(Forest\_biodiversity\_data)[4] = "basal\_area"

Forest\_biodiversity\_data$block = unlist(strsplit(Forest\_biodiversity\_data$block, split = " "))

Forest\_biodiversity\_data$basal\_area = as.numeric(Forest\_biodiversity\_data$basal\_area)

Forest\_biodiversity\_data[which(is.na(Forest\_biodiversity\_data$basal\_area))[1], 'basal\_area'] =

mean(Forest\_biodiversity\_data[Forest\_biodiversity\_data$block == "Germany" & Forest\_biodiversity\_data$full\_species\_original == "Quercus.robur", "basal\_area"], na.rm = T)

Forest\_biodiversity\_data[which(is.na(Forest\_biodiversity\_data$basal\_area)), 'basal\_area'] =

mean(Forest\_biodiversity\_data[Forest\_biodiversity\_data$block == "Germany" & Forest\_biodiversity\_data$full\_species\_original == "Acer.pseudoplatanus", "basal\_area"], na.rm = T)

for (x in c("FIN02", "FIN03", "FIN04", "FIN07", "FIN08", "FIN12",

"FIN13", "FIN15", "FIN20", "FIN24", "FIN25", "FIN26", "FIN28")) {

Forest\_biodiversity\_data[Forest\_biodiversity\_data$plot == x & Forest\_biodiversity\_data$full\_species\_original == "Betula.pendula", "basal\_area"] =

sum(Forest\_biodiversity\_data[Forest\_biodiversity\_data$plot == x & Forest\_biodiversity\_data$full\_species\_original %in% c("Betula.pendula", "Betula.pubescens"),

"basal\_area"])

Forest\_biodiversity\_data = Forest\_biodiversity\_data[ -which(Forest\_biodiversity\_data$plot == x &

Forest\_biodiversity\_data$full\_species\_original == "Betula.pubescens"),]

}

Forest\_biodiversity\_data[Forest\_biodiversity\_data$plot == "GER05" & Forest\_biodiversity\_data$full\_species\_original == "Quercus.petraea", "basal\_area"] =

sum(Forest\_biodiversity\_data[Forest\_biodiversity\_data$plot == "GER05" & Forest\_biodiversity\_data$full\_species\_original %in% c("Quercus.petraea", "Quercus.robur"), "basal\_area"])

Forest\_biodiversity\_data = Forest\_biodiversity\_data[ -which(Forest\_biodiversity\_data$plot == "GER05"

& Forest\_biodiversity\_data$full\_species\_original == "Quercus.robur"),]

Forest\_biodiversity\_data$full\_species\_original[Forest\_biodiversity\_data$full\_species\_original == "Betula.pubescens"] = "Betula.pendula"

Forest\_biodiversity\_data$full\_species\_original[Forest\_biodiversity\_data$full\_species\_original == "Quercus.robur"] = "Quercus.petraea"

Second, use function ‘function.normalization’ (provided in the source code) to transform the raw function data (‘Forest\_function\_data\_raw’) to [0, 1] within each country. Then, the correlation and distance of any two functions are obtained based on the transformation .

variables = colnames(Forest\_function\_data\_raw)[-(1:5)]

Forest\_function\_data\_raw = function.normalization(Forest\_function\_data\_raw)

variables.std <- paste0(variables, ".std")

correlation = Forest\_function\_data\_raw[,variables.std]

correlation = cor(correlation)

distM = sqrt(1 - abs(correlation))

To plot Figure 2, the two functions ‘MF.uncor’, ‘MF.cor’ (provided in the source code) are first used to compute multifunctionality of order *q* = 0, 1, 2 when the correlation between any two functions is corrected for (function ‘MF.cor’) or not (function ‘MF.uncor’). Then the function ‘qD’ is used to compute species diversity of order *q* = 0, 1, 2 for 209 plots (based on the biodiversity data ‘Forest\_biodiversity\_data’). In addition, the function ‘lmer’ in the package ‘lmerTest’ are adopted to model the relationship between tree species diversity of order q and within-plot multifunctionality values of the same order using a linear mixed-effects model with random slopes and random intercepts for each country.

**## Compute multifunctionality**

Forest\_function\_data\_raw <- Forest\_function\_data\_raw %>%

mutate(mf\_Chao\_0 = apply(Forest\_function\_data\_raw[,variables.std], 1, function(x)

MF.uncor(x, rep(1, length(x)), 0)$qMF),

mf\_Chao\_1 = apply(Forest\_function\_data\_raw[,variables.std], 1, function(x)

MF.uncor(x, rep(1, length(x)), 1)$qMF),

mf\_Chao\_2 = apply(Forest\_function\_data\_raw[,variables.std], 1, function(x)

MF.uncor(x, rep(1, length(x)), 2)$qMF),

mf\_Chao\_AUC\_0 = apply(Forest\_function\_data\_raw[,variables.std], 1, function(x)

MF.cor(x, rep(1, length(x)), distM, q = 0) %>% filter(tau == 'AUC') %>% select(qMF) %>% as.numeric),

mf\_Chao\_AUC\_1 = apply(Forest\_function\_data\_raw[,variables.std], 1, function(x)

MF.cor(x, rep(1, length(x)), distM, q = 1) %>% filter(tau == 'AUC') %>% select(qMF) %>% as.numeric),

mf\_Chao\_AUC\_2 = apply(Forest\_function\_data\_raw[,variables.std], 1, function(x)

MF.cor(x, rep(1, length(x)), distM, q = 2) %>% filter(tau == 'AUC') %>% select(qMF) %>% as.numeric)

)

mf = Forest\_function\_data\_raw %>%

select(plotid, target\_species\_richness, mf\_Chao\_0:mf\_Chao\_AUC\_2) %>%

pivot\_longer(cols = c(mf\_Chao\_0:mf\_Chao\_AUC\_2), names\_to = "method") %>%

mutate(method = fct\_inorder(method))

**## Compute species diversity**

mf = mf %>% mutate(target\_species\_richness = sapply(unique(Forest\_biodiversity\_data$plot), function(x)

rep( qD( (Forest\_biodiversity\_data %>% filter(plot == x))$basal\_area, q = c(0,1,2)), 2)) %>% as.vector)

mf = mf %>% mutate(Order.q = rep(c(0, 1, 2), 2\*nrow(Forest\_function\_data\_raw)),

Div = rep(rep(c('Uncorrected', 'Corrected'), each = 3), nrow(Forest\_function\_data\_raw)))

**## Fit linear mixed model**

mf = mf %>% arrange(method)

mf = mf %>% group\_by(method) %>%

do(lmer(formula = value ~ 1 + target\_species\_richness + (1 + target\_species\_richness | plotid),

data = . ) %>% predict %>% tibble(fit = .)) %>%

ungroup %>% select(fit) %>% bind\_cols(mf) %>%

mutate("p\_value" = 0, 'Sig' = "Significant slope (P < 0.05)")

mf$Order.q = as.factor(mf$Order.q)

slope = lapply(1:length(unique(mf$method)), function(j) {

myout\_ <- mf %>% filter(method == unique(mf$method)[j])

model = lmer(formula = value ~ 1 + target\_species\_richness + (1 + target\_species\_richness | plotid),

data = myout\_)

##

slope = coef(model)$plotid$target\_species\_richness

slope = ifelse(round(abs(slope), 3) < 0.001, slope %>% round(., 4) %>%

paste("Slope = ", ., sep = ""), ifelse(round(abs(slope), 2) < 0.01, slope %>% round(., 3) %>%

paste("Slope = ", ., sep = ""), format(slope %>% round(., 2), nsmall = 2) %>%

paste("Slope = ", ., sep = "")))

tmp = myout\_[ !duplicated(myout\_[, c('plotid', 'Order.q', 'Div')]),] %>%

select(c('plotid', 'Order.q', 'Div'))

##

slope\_lmm = summary(model)$coefficients[2, 'Estimate']

slope\_lmm = ifelse(round(abs(slope\_lmm), 3) < 0.001, slope\_lmm %>% round(., 4) %>%

paste("slope = ", ., sep = ""),

ifelse(round(abs(slope\_lmm), 2) < 0.01, slope\_lmm %>% round(., 3) %>%

paste("slope = ", ., sep = ""), format(slope\_lmm %>% round(., 2), nsmall = 2) %>%

paste("slope = ", ., sep = "")))

tmp\_lmm = myout\_[ !duplicated(myout\_[, c('Order.q', 'Div')]),] %>% select(c('Order.q', 'Div'))

##

rbind(cbind(tmp, x = max(myout\_$target\_species\_richness), y = max(myout\_$value), Slope = slope),

cbind(plotid = 'Linear mixed', tmp\_lmm, x = max(myout\_$target\_species\_richness),

y = max(myout\_$value), Slope = slope\_lmm))

}) %>% do.call(rbind,.)

slope = slope %>% mutate(plotid = fct\_inorder(plotid)) %>% arrange(plotid)

mf = rbind(mf,

lapply(unique(mf$method), function(x) {

lmm.data = mf %>% filter(method == x)

model <- lmer(formula = value ~ 1 + target\_species\_richness +

(1 + target\_species\_richness | plotid), data = lmm.data)

data.frame(fit = predict(model, re.form = NA),

plotid = 'Linear mixed',

target\_species\_richness = lmm.data$target\_species\_richness,

value = 0,

method = x,

Order.q = unique(lmm.data$Order.q),

Div = unique(lmm.data$Div),

p\_value = summary(model)$coefficients[2, 'Pr(>|t|)'],

Sig = ifelse( summary(model)$coefficients[2, 'Pr(>|t|)'] < 0.05,

'Significant slope (P < 0.05)', 'Nonsignificant slope' ))

}) %>% do.call(rbind,.)

)

Finally, run the following code to plot Figure 2. This figure also shows the overall fixed-effect slopes and each country’s relationships estimated from the same linear mixed model.

fig.theme = theme(legend.position = "bottom",

legend.box = "vertical",

legend.key.width = unit(1.2, "cm"),

text = element\_text(size = 16),

plot.margin = unit(c(5.5, 5.5, 5.5, 5.5), "pt"),

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

guide = guides(colour = guide\_legend(title = "Plot ID", override.aes = list(linewidth = 1.5)),

lty = guide\_legend(override.aes = list(linewidth = 1, colour = "red")),

size = "none")

col\_manual = scale\_colour\_manual(breaks = c("FIN", "GER", "ITA", "POL",

"ROM", "SPA", "Linear mixed"),

label = c("Finland (3)", "Germany (5)", "Italy (5)",

"Poland (5)", "Romania (4)", "Spain (4)", "Linear mixed"),

values = c("FIN" = "black", "GER" = "purple2", "ITA" = "darkorange",

"POL" = "steelblue1", "ROM" = "blue", "SPA" = "gray55", "Linear mixed" = "red"))

size\_manual = scale\_size\_manual(values = c("FIN" = 0.5, "GER" = 0.5, "ITA" = 0.5, "POL" = 0.5,

"ROM" = 0.5, "SPA" = 0.5, "Linear mixed" = 1.9))

lty\_manual = scale\_linetype\_manual(values = c("Nonsignificant slope" = "dashed",

"Significant slope (P < 0.05)" = "solid"), name = NULL, drop = FALSE)

ggplot() +

geom\_line(data = mf %>% mutate(Div = fct\_inorder(Div)),

aes(x = target\_species\_richness, y = fit, col = plotid, lty = Sig, size = plotid)) +

geom\_point(data = mf %>% filter(plotid != 'Linear mixed') %>%

mutate(Div = fct\_inorder(Div)),

aes(x = target\_species\_richness, y = value, color = plotid), alpha = 0.2) +

geom\_text(data = slope %>% mutate(Div = fct\_inorder(Div),

x = max(x) - rep(c(3, 1, 3, 1, 3, 1, 3), each = 6),

y = max(y) + c(rep(0.7, 12), rep(0.2, 12), rep(-0.3, 12), rep(-0.8, 6))),

aes(x = x, y = y, label = Slope, color = plotid), size = 5, show.legend = FALSE) +

theme\_bw() +

facet\_grid(Div ~ Order.q, scale = "fixed",

labeller = labeller(Order.q = c(`0` = "q = 0", `1` = "q = 1", `2` = "q = 2"),

Div = c(`Uncorrected` = "Uncorrected for correlations",

`Corrected` = "Corrected for correlations"))) +

labs(x = "Species diversity", y = "Multifunctionality") +

col\_manual +

size\_manual +

lty\_manual +

fig.theme +

guide +

coord\_cartesian(ylim = c(7.3, 15.2))

##################################################################################

1. **Figure 3 and Figure 4: Biodiversity-multifunctionality relationships across spatial scales when the correlation between any two functions is not corrected for (Figure 3) or corrected for (Figure 4).**

The two functions ‘beta.MF.uncor’ (uncorrected for correlations) and ‘beta.MF.cor’ (corrected for correlations) are used to compute alpha, beta, and gamma multifunctionality of order *q* = 0, 1, 2, based on any two chosen plots within each of the five European countries (Finland was excluded). The total number of pairwise combinations is 703 (Germany) + 630 (Italy) + 903 (Poland) + 378 (Romania) + 630 (Spain) = 3244. At the same time, also compute corresponding alpha, beta, and gamma species diversity of order *q* = 0, 1, 2 based on any two chosen plots within each country (based on ‘Forest\_biodiversity\_data’ data).

Note: We use the function ‘parLapply’ in the package ‘parallel’ to accelerate the computation.

cpu.cores <- detectCores()-1

cl <- makeCluster(cpu.cores)

clusterExport(cl, varlist = c("Forest\_function\_data\_raw", "variables.std", "distM", "qD", "MF.uncor", "MF.cor", "beta.MF.uncor", "beta.MF.cor", "Forest\_biodiversity\_data"),

envir = environment())

clusterEvalQ(cl, c(library(dplyr), library(tidyr), library(reshape2)))

beta.result.uncor = parLapply(cl, unique(Forest\_function\_data\_raw$plotid), function(x) {

country.data = Forest\_function\_data\_raw %>% filter(plotid == x)

comb = combn(1:nrow(country.data), 2)

species.data = Forest\_biodiversity\_data[substr(Forest\_biodiversity\_data$plot, 1, 3) == x,]

lapply(1:ncol(comb), function(i) {

index = comb[,i]

data = country.data[index, ]

N = length(index)

Species = species.data %>% filter(plot %in% unique(species.data$plot)[index]) %>%

acast(., full\_species\_original ~ plot, value.var = "basal\_area")

Species[is.na(Species)] = 0

Gamma.div = qD(rowSums(Species), q = c(0,1,2))

Alpha.div = qD(as.vector(Species), q = c(0,1,2)) / N

Beta.div = Gamma.div / Alpha.div

out = beta.MF.uncor(data[,variables.std] %>% t, q = c(0,1,2))

out %>% filter(Type %in% c('Gamma','Alpha','Beta')) %>%

mutate(Gamma = rep(Gamma.div, 3), Alpha = rep(Alpha.div, 3), Beta = rep(Beta.div, 3)) %>%

pivot\_longer(cols = c(Gamma:Beta), names\_to = 'Dissimilarity')

}) %>% do.call(rbind,.) %>% mutate(plotid = x)

}) %>% do.call(rbind,.)

beta.result.cor = parLapply(cl, unique(Forest\_function\_data\_raw$plotid), function(x) {

country.data = Forest\_function\_data\_raw %>% filter(plotid == x)

comb = combn(1:nrow(country.data), 2)

species.data = Forest\_biodiversity\_data[substr(Forest\_biodiversity\_data$plot, 1, 3) == x,]

lapply(1:ncol(comb), function(i) {

index = comb[,i]

data = country.data[index, ]

N = length(index)

Species = species.data %>% filter(plot %in% unique(species.data$plot)[index]) %>%

acast(., full\_species\_original ~ plot, value.var = "basal\_area")

Species[is.na(Species)] = 0

Gamma.div = qD(rowSums(Species), q = c(0,1,2))

Alpha.div = qD(as.vector(Species), q = c(0,1,2)) / N

Beta.div = Gamma.div / Alpha.div

out = beta.MF.cor(data[,variables.std] %>% t, distM = distM, q = c(0,1,2)) %>%

filter(tau == 'AUC') %>% select(-tau)

out %>% filter(Type %in% c('Gamma','Alpha','Beta')) %>%

mutate(Gamma = rep(Gamma.div, 3), Alpha = rep(Alpha.div, 3), Beta = rep(Beta.div, 3)) %>%

pivot\_longer(cols = c(Gamma:Beta), names\_to = 'Dissimilarity')

}) %>% do.call(rbind,.) %>% mutate(plotid = x)

}) %>% do.call(rbind,.)

stopCluster(cl)

To plot Figure 3, use the function ‘fig\_alpha\_gamma\_beta’ (provided in the source code) to obtain (a) alpha multifunctionality vs. alpha species diversity, (b) beta multifunctionality vs. beta species diversity, and (c) gamma multifunctionality vs. gamma species diversity, of order *q* = 0, 1, 2 when the correlation between any two functions is not incorporated. For each scale, the relationship between alpha/beta/gamma multifunctionality values of order q and diversity of the same order was modeled using a linear mixed-effects model with random slopes and random intercepts for each country. Figure 3 also shows the overall fixed-effect slopes and each country’s relationships estimated from the same linear mixed model.

fig3 = fig\_alpha\_gamma\_beta(beta.result.uncor, type = "uncorrected")

fig3$fig.alpha ## figure 3a

fig3$fig.beta ## figure 3b

fig3$fig.gamma ## figure 3c

In a similar manner, use the function ‘fig\_alpha\_gamma\_beta’ but the correlation between any two functions is incorporated. Run the following code to obtain Figure 4.

fig4 = fig\_alpha\_gamma\_beta(beta.result.cor, type = "corrected")

fig4$fig.alpha ## figure 4a

fig4$fig.beta ## figure 4b

fig4$fig.gamma ## figure 4c