Kutchartt E, Gayoso J, Pirotti F, Bucarey Á, Guerra J, Hernández J, Corvalán P, Drápela K, Olson M, Zwanzig M (2021). **Aboveground tree biomass of *Araucaria araucana* in southern Chile: measurements and multi-objective optimization of biomass models**
iForest – Biogeosciences and Forestry – doi: 10.3832/ifor3492-013

Supplementary Material

Fig. S1 - Destructive sampling, including all measurements before and after felling trees (left) and samples taken to determine basic density and moisture content at different stem positions (right).

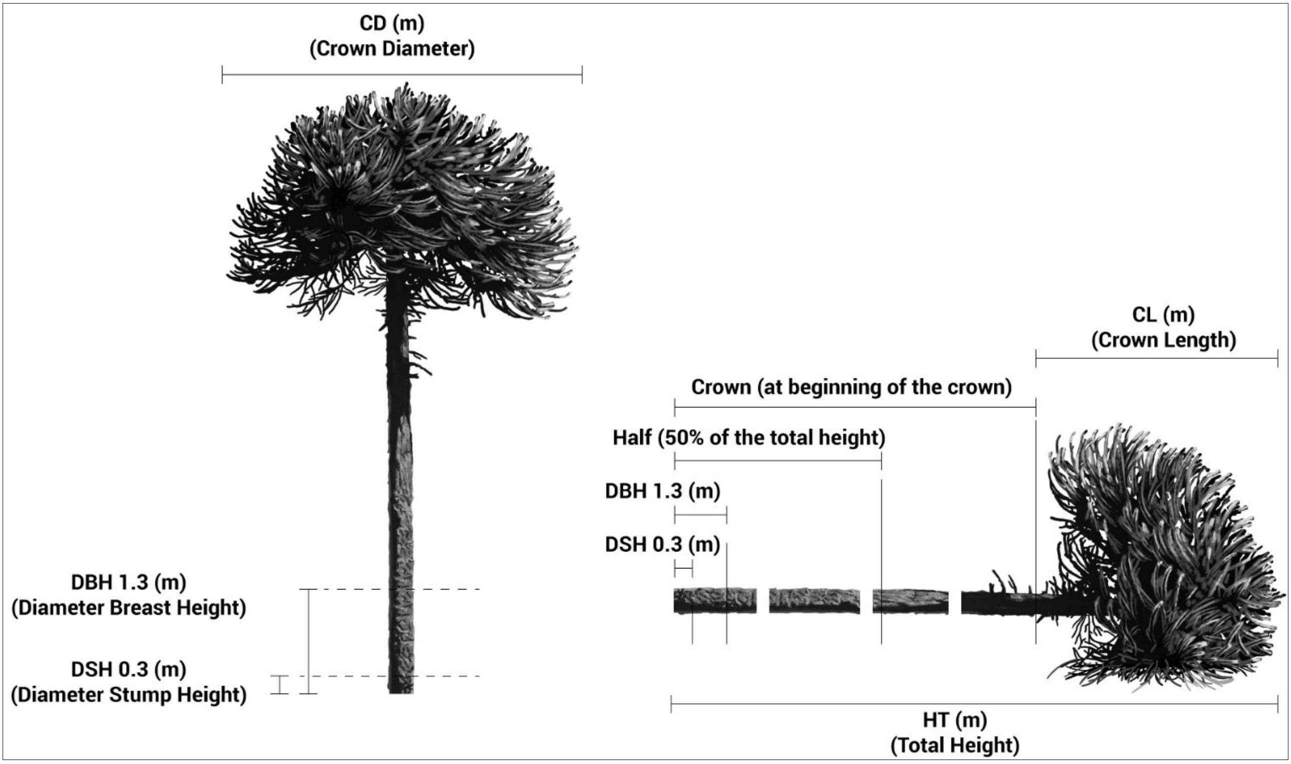


Fig. S2 - Foliage component. Representation of foliage component, which includes branches and scaly, leathery leaves as a whole component.

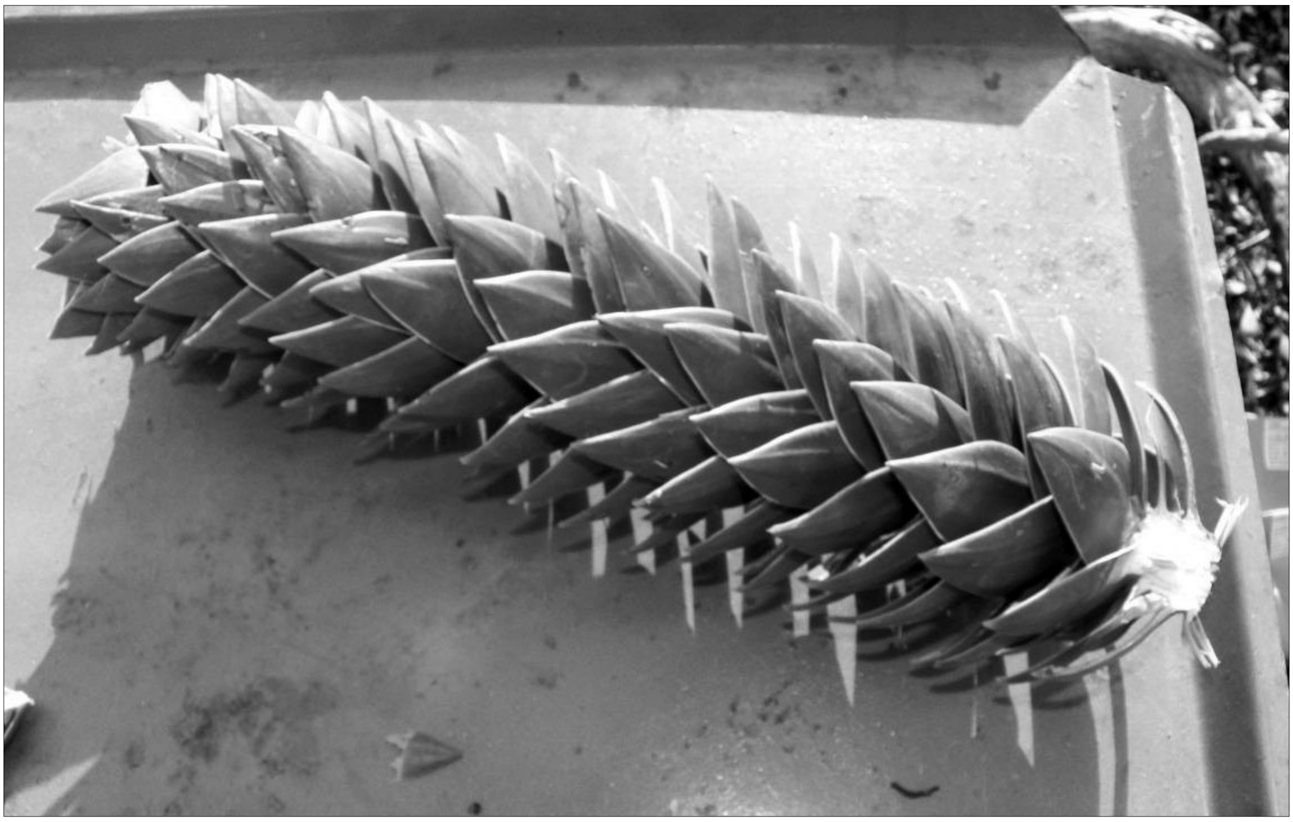


Fig. S3 - Scatter graph for the relation of aboveground biomass (kg tree^{-1}) to DBH (cm): (A) whole tree, (B) stem wood, (C) stem bark and (D) foliage.

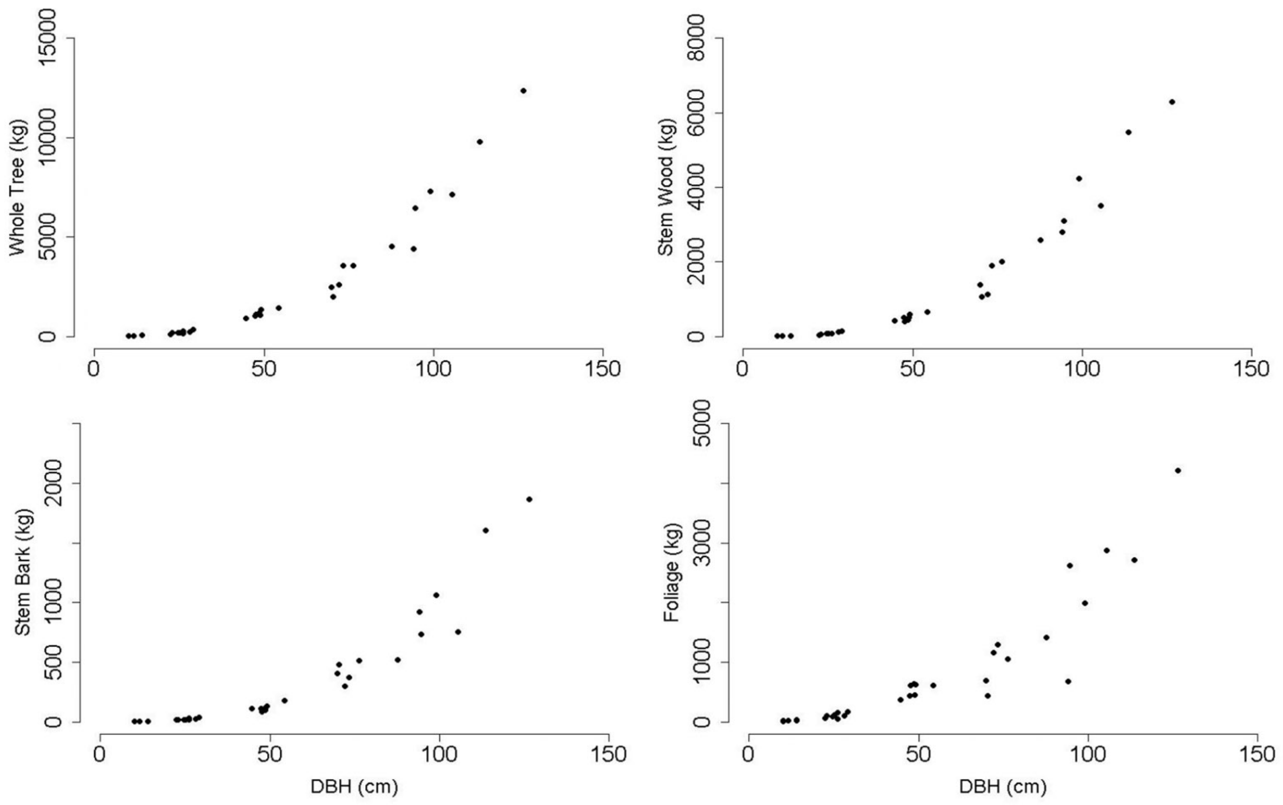
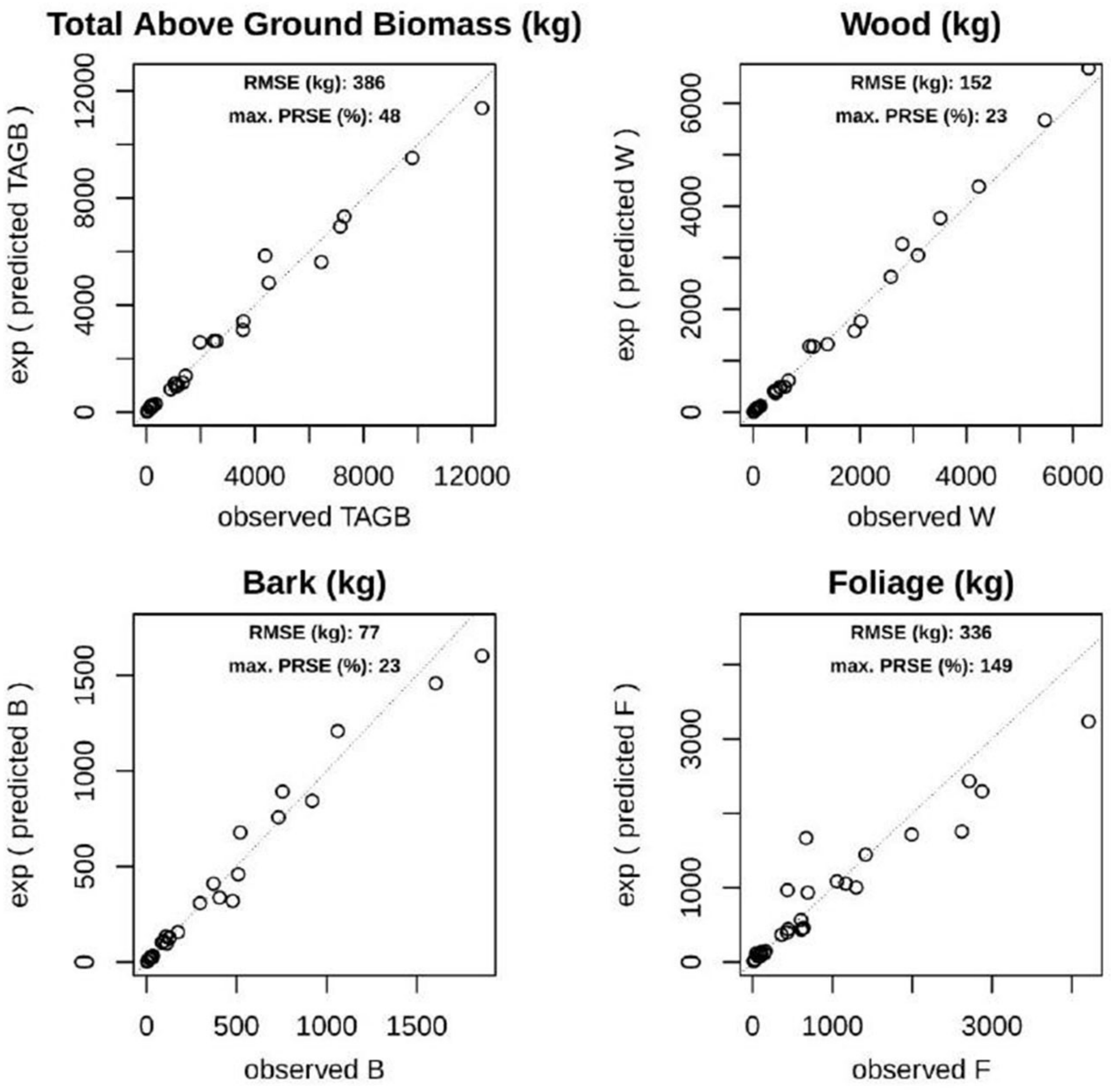


Fig. S4 - Model prediction in the equation 9 using predicted versus observed values for tree components and total aboveground biomass.



Appendix 1 - R-script for multi-objective model comparison.

```
# R-Script for multi-objective model comparison
# by Martin Zwanzig (https://orcid.org/0000-0003-1866-6743)
# based on R version 3.6.3
# Citations of the aligned article are welcome:
# Kutchartt et al. 2021. "Above ground tree biomass of Araucaria araucana in southern Chile
# ©measurements and multi-objective optimization of biomass models"

library(caret)
library(ggplot2)

# LOAD DATA ----
b <- read.csv2(file.choose()) # "Database.csv"

b$AGE <- b$Age..years. # Let's give a short variable name

# Calculate some additional variables that will be used as covariates in the models:
b$HTtoDBH <- b$HT / b$DBH*100 # h/d-value, e.g. used as indicator for the stability of a tree
(especially for spruce plantations)
b$DBH2HT <- b$DBH^2 * b$HT
b$DBHHT2 <- b$DBH * b$HT^2

# CROSS VALIDATION STATISTICS FOR DIFFERENT MODELS AND RESPONSE VARIABLES ----

#Define some functions that are used (besides others) to calculate model performance measures
# percent relative standard error (PRSE)
PRSE <- function(coeff.sd, coeff.mean) {100 * coeff.sd / coeff.mean}
# root mean square error (RMSE)
RMSE <- function(fit, obs) {sqrt(sum((fit - obs)^2) / length(obs))} # using fit and obs on
transformed scale
# mean absolute percentage error (MAPE %)
MAPE <- function(fit, obs) {(100 / length(obs)) * sum(abs(obs- fit) / obs)} # using back-transformed
predicted values; obs on original scale
# bias % (comparable between different models)
biasPERCENT <- function(fit, obs) {(100 / length(obs)) * sum((fit - obs) / obs)} # using back-
transformed predicted values; obs on original scale
# bias as the mean of model residuals (not comparable between different models)
biasMEANRES <- function(fit, obs) {mean(fit - obs)} # using back-transformed predicted values; obs
on original scale

# define a function which can be used for a repeated analysis for different models:
m.stats <- function(form, obs, response.name, log.response){
  m.def <- lm(form, data = obs) # default linear model
  m.LOOCV <- train(form, obs, method="lm", # Leave one out cross validation
                 trControl=trainControl(method="LOOCV"))
  m.kfold <- train(form, obs, method="lm", # K-foild cross validation
                 trControl=trainControl(method = "repeatedcv",
                                        number = 7, repeats = 20))
  # consider if logarithmic values were used or not:
  if(log.response == TRUE) { RMSEobs <- log(obs[,response.name])}
  if(log.response == FALSE){ RMSEobs <- obs[,response.name]}
  if(log.response == TRUE) { predorig <- exp(m.def$fitted.values)}
  if(log.response == FALSE){ predorig <- m.def$fitted.values}
  NA.n <- rep(NA, 17 - length(coef(m.def))) # when models have more parameters, this 'empty space'
gets filled
  def.stats <- as.data.frame(cbind( # summary of core statistics of the default model (simple linear
model)
                     summary(m.def)$r.squared,
                     AIC(m.def),
                     RMSE(fit = predorig, obs = obs[,response.name]), # using fit and obs on
ORIGINAL scale (DIFFERENCE TO PREVIOUS CALCULATIONS)
                     MAPE(fit = predorig, obs = obs[,response.name]), # using back-transformed
predicted values; obs on original scale
                     biasPERCENT(fit = predorig, obs = obs[,response.name]), # using back-
transformed predicted values; obs on original scale
                     biasMEANRES(fit = predorig, obs = obs[,response.name]), # using back-
transformed predicted values; obs on original scale
                     rbind(c(coef(m.def), NA.n)),
                     rbind(c(PRSE(coeff.mean = coef(m.def), coeff.sd = summary(m.def)
$coefficients[,2]), NA.n))
  ))
  colnames(def.stats) <- c("R2", "AIC","RMSEkg", "MAPE","biasPERCENT","biasMEANRES",
                           rep("coeff", 17),rep("PRSE", 17))
  LOOCV.stats <- cbind(m.LOOCV$results, rbind(c(coef(m.LOOCV$finalModel), NA.n))) # summary for
LOOCV
  names(LOOCV.stats)[5:21] <- rep("coeff", 17)
  kfold.stats <- cbind(m.kfold$results, rbind(c(coef(m.kfold$finalModel), NA.n))) # summary for K-
fold CV
  names(kfold.stats)[8:24] <- rep("coeff", 17)
  return(list( # merge all calculated statistics in one R object
    allobs = def.stats,
    LOOCV = LOOCV.stats,
    kfold = kfold.stats
  )
  )
}

# example:
m.stats(form = log(TAGB) ~ log(DBH), obs = b, response.name = "TAGB", log.response = TRUE)
# NOTE: when models have more parameters, the space with 'NA' gets filled;
#       ... we just need rows of equal length for each model evaluation in the following

# first set of covariate functions with transformed response:
covariate_functions <- c("log(DBH)",
                         "log(HT)",
                         "log(CL)",
                         "log(CD)",
                         "log(DBH) + HT",
                         "log(DBH) + CL",
                         "log(DBH) + CD",
                         "log(DBH) + HT + CL",
                         "log(DBH) + HT + CD",
                         "log(DBH) + CL + CD",
                         "log(DBH) + HT + CL + CD",
                         "log(DBH) * HT",
                         "log(DBH) * CL",
                         "log(DBH) * CD",
                         "log(DBH) * HT * CL",
                         "log(DBH) * HT * CD",
                         "log(DBH) * CL * CD",
                         "log(DBH) * HT * CL * CD",
                         "log(DBH) + log(HT)",
                         "log(DBH) + log(CL)",
                         "log(DBH) + log(CD)",
                         "log(DBH) + log(HT) + log(CL)",
                         "log(DBH) + log(HT) + log(CD)",
                         "log(DBH) + log(CL) + log(CD)",
                         "log(DBH) + log(HT) + log(CL) + log(CD)",
                         "log(DBH) * log(HT)",
                         "log(DBH) * log(CL)",
                         "log(DBH) * log(CD)",
                         "log(DBH) * log(HT) * log(CL)",
                         "log(DBH) * log(HT) * log(CD)",
                         "log(DBH) * log(CL) * log(CD)",
                         "log(DBH) * log(HT) * log(CL) * log(CD)")
# second set of covariate functions with non-transformed response:
covariate_functions2 <- c("DBH",
                          "DBH + I(HT^2)",
                          "DBH + I(CL^2)",
                          "DBH + I(CD^2)",
                          "DBH + I(HT^2) + I(CL^2)",
                          "DBH + I(HT^2) + I(CD^2)",
                          "DBH + I(CD^2) + I(CL^2)",
                          "DBH + I(HT^2) + I(CL^2) + I(CD^2)",
                          "I(DBH^2)",
                          "I(DBH^2) + I(HT^2)",
                          "I(DBH^2) + I(CL^2)",
                          "I(DBH^2) + I(CD^2)",
                          "I(DBH^2) + I(HT^2) + I(CL^2)",
                          "I(DBH^2) + I(HT^2) + I(CD^2)",
                          "I(DBH^2) + I(CD^2) + I(CL^2)",
                          "I(DBH^2) + I(HT^2) + I(CL^2) + I(CD^2)",
                          "DBH + DBH2HT",
                          "DBH + DBHHT2",
                          "I(DBH^2) + DBH2HT",
                          "I(DBH^2) + DBHHT2",
                          "I(DBH^2) + DBH2HT + DBHHT2",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(HT^2)",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(CD^2)",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(CL^2)",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(CL^2) + I(CD^2)",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(HT^2) + I(CL^2)",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(CD^2) + I(HT^2)",
                          "I(DBH^2) + DBH2HT + DBHHT2 + I(CD^2) + I(HT^2) + I(CL^2)",
                          "I(DBH^2) + DBH2HT + I(HT^2)",
                          "I(DBH^2) + DBH2HT + I(CD^2)",
                          "I(DBH^2) + DBH2HT + I(CL^2)",
                          "I(DBH^2) + DBH2HT + I(CL^2) + I(CD^2)",
                          "I(DBH^2) + DBH2HT + I(HT^2) + I(CL^2)",
                          "I(DBH^2) + DBH2HT + I(CD^2) + I(HT^2)",
                          "I(DBH^2) + DBH2HT + I(CD^2) + I(HT^2) + I(CL^2)",
                          "I(DBH^2) + DBHHT2 + I(HT^2)",
                          "I(DBH^2) + DBHHT2 + I(CD^2)",
                          "I(DBH^2) + DBHHT2 + I(CL^2)",
                          "I(DBH^2) + DBHHT2 + I(CL^2) + I(CD^2)",
                          "I(DBH^2) + DBHHT2 + I(HT^2) + I(CL^2)",
                          "I(DBH^2) + DBHHT2 + I(CD^2) + I(HT^2)",
                          "I(DBH^2) + DBHHT2 + I(CD^2) + I(HT^2) + I(CL^2)",
                          "DBH + I(DBH^2) + I(CL^2) + I(CD^2)"
)
obs <- b # define 'obs' as a copy of our dataset. It is used in the loop function that follows
res.allobs <- NULL; res.LOOCV <- NULL; res.kfold <- NULL # create empty variables
set.seed(123) # define a seed to make the following results reproducible
for (j in 1:4){
  response.name <- c("TAGB", "W" , "B", "F")[j] # do it for every response variable
  # first set of covariate functions with transformed response:
  log.response <- TRUE
  for (i in 1:length(covariate_functions)){
    response <- ifelse(log.response, paste("log(",response.name,")",sep =""), response.name) # log
transformation or not
    covariates <- covariate_functions[i] # evaluate each covariate function
    form <- as.formula(paste(response, " ~ ", covariates, sep = "")) # formula
    print(form) # show the formula that is going to be evaluated
    DBH <- sum(grep("DBH",covariates)>0)>0 # is DBH part of the formula?
    CL <- sum(grep("CL",covariates)>0)>0   # is CL part of the formula?
    HT <- sum(grep("HT",covariates)>0)>0   # is HT part of the formula?
    CD <- sum(grep("CD",covariates)>0)>0   # is CD part of the formula?
    fstats <- m.stats(form = form, obs = obs, response.name = response.name, log.response =
log.response)
    res.allobs <- rbind(res.allobs, cbind(log.response, response.name, response, covariates, DBH,
CL, HT, CD, fstats$allobs))
    res.LOOCV <- rbind(res.LOOCV, cbind(log.response, response.name, response, covariates, DBH, CL,
HT, CD, fstats$LOOCV))
    res.kfold <- rbind(res.kfold, cbind(log.response, response.name, response, covariates, DBH, CL,
HT, CD, fstats$kfold))
  }
  # second set of covariate functions with non-transformed response:
  log.response <- FALSE
  for (i in 1:length(covariate_functions2)){
    response <- ifelse(log.response, paste("log(",response.name,")",sep =""), response.name) # log
transformation or not
    covariates <- covariate_functions2[i] # evaluate each covariate function
    form <- as.formula(paste(response, " ~ ", covariates, sep = "")) # formula
    print(form) # show the formula that is going to be evaluated
    DBH <- sum(grep("DBH",covariates)>0)>0 # is DBH part of the formula?
    CL <- sum(grep("CL",covariates)>0)>0   # is CL part of the formula?
    HT <- sum(grep("HT",covariates)>0)>0   # is HT part of the formula?
    CD <- sum(grep("CD",covariates)>0)>0   # is CD part of the formula?
    fstats <- m.stats(form = form, obs = obs, response.name = response.name, log.response = 
log.response)
    res.allobs <- rbind(res.allobs, cbind(log.response, response.name, response, covariates, DBH, 
CL, HT, CD, fstats$allobs))
    res.LOOCV <- rbind(res.LOOCV, cbind(log.response, response.name, response, covariates, DBH, CL, 
HT, CD, fstats$LOOCV))
    res.kfold <- rbind(res.kfold, cbind(log.response, response.name, response, covariates, DBH, CL, 
HT, CD, fstats$kfold))
  }
}
# calculate maximum PRSE for each model (highest uncertainty of one of the estimated model 
parameters)
res.allobs$PRSEmax <- as.numeric(apply(res.allobs[,32:48], MARGIN = 1, FUN = function(x)
{max(abs(na.exclude(x)))}))
# make a local copy of the results, if you wish
write.csv(res.allobs, "allobs_comparison.csv")
write.csv(res.LOOCV, "LOOCV_comparison.csv")
write.csv(res.kfold, "kfold_comparison.csv")

# SELECT models with maximum PRSE BELOW critical threshold (either 20, 25, 30 or 50)
PRSE30set <- subset(res.allobs[c(3,4,9:14,49)], res.allobs$PRSEmax < 30) # using 30, enough models 
for further comparison are left
write.csv(PRSE30set, "PRSEmax30.csv") # a set of models with low uncertainty in model parameters

# MODEL SELECTION BASED ON MULTI-OBJECTIVE OPTIMIZATION
# this aims to find those models with best model performance
# based on more than one performance measure, e.g. high R², low PRSEmax, low MAPE;
# to make performance measures comparable, they are scaled for each response variable:
scale_R2 <- tapply(1/res.kfold$Rsquared, INDEX = res.kfold$response.name, FUN = scale)
scale_RMSE <- tapply(res.kfold$RMSE, INDEX = res.kfold$response.name, FUN = scale)
scale_PRSEmax <- tapply(res.allobs$PRSEmax, INDEX = res.allobs$response.name, FUN = scale)
scale_MAPE <- tapply(res.allobs$MAPE, INDEX = res.allobs$response.name, FUN = scale)
scale_RMSEkg <- tapply(res.allobs$RMSEkg, INDEX = res.allobs$response.name, FUN = scale)
scale_R2a <- tapply(1/res.allobs$R2, INDEX = res.allobs$response.name, FUN = scale)

# TO GET AN OVERALL PERFORMANCE FOR EVERY COVARIATE FUNCTION, SCALED MEASURES HAVE TO BE SUMMED UP
# First, matrix is defined for each list of scaled measures to be able to perform some matrix 
operations later on:
scale_R2m <- cbind(scale_R2[[c("TAGB")]],scale_R2[[c("W")]],scale_R2[[c("B")]],scale_R2[[c("F")]] )
scale_RMSEm <- 
cbind(scale_RMSE[[c("TAGB")]],scale_RMSE[[c("W")]],scale_RMSE[[c("B")]],scale_RMSE[[c("F")]] )
scale_PRSEmaxm <- 
cbind(scale_PRSEmax[[c("TAGB")]],scale_PRSEmax[[c("W")]],scale_PRSEmax[[c("B")]],scale_PRSEmax[[c("F
")]] )
scale_MAPEm <- 
cbind(scale_MAPE[[c("TAGB")]],scale_MAPE[[c("W")]],scale_MAPE[[c("B")]],scale_MAPE[[c("F")]] )
scale_RMSEkgm <- 
cbind(scale_RMSEkg[[c("TAGB")]],scale_RMSEkg[[c("W")]],scale_RMSEkg[[c("B")]],scale_RMSEkg[[c("F")]]
)
scale_R2am <- 
cbind(scale_R2a[[c("TAGB")]],scale_R2a[[c("W")]],scale_R2a[[c("B")]],scale_R2a[[c("F")]] )

# Only a subset of criteria is used
# (Otherwise the specific effect of criteria that are
#  correlated might be weighted to strong)
scalesum <- 
  scale_R2m + # from k-fold cross validation
  #scale_RMSEm + # from k-fold cross validation
  scale_PRSEmaxm + # from fit to all observations
  scale_MAPEm #+ # from fit to all observations
  #scale_R2am #+ # from fit to all observations
  #scale_RMSEkgm # from fit to all observations

scalesum

# calculation of scale sums for every covariate function
# (depending on RÂ² of CV, PRSEmax and MAPE performance on each response)
csumt <- as.data.frame(scalesum)
colnames(csumt) <- c("TAGB","W","B","F")
csumt <- cbind(res.kfold[1:(length(covariate_functions)+length(covariate_functions2)),4:8],csumt)
```

```
head(csumt)
csumt$scalesum <- rowSums(csumt[,c("TAGB","W","B","F")])
csumt$scalesum_rank <- rank(csumt$scalesum, ties.method= "average")

# Order the ranksums, lowest ranksums indicate best performance of a
# covariate function on all performance measures for all response variables:
orderedlist <- csumt[order(csumt$scalesum_rank),]
head(orderedlist)
# Save ordered list:
write.csv(orderedlist, "scalesum_rank_orderedlist.csv")

ord <- as.numeric(rownames(orderedlist))
ord

# Combine all performance measures and estimated coefficients for the "Top10"
# of models (according to their ranksum):
all_resp_top10_rank <- rbind(
cbind(subset(res.allobs, res.allobs$response.name == "TAGB")[,c(3,4,15:19,32:36,49,9:14)],
      subset(res.kfold, res.kfold$response.name == "TAGB")[,c(10:15)])[ord[1:10],],
cbind(subset(res.allobs, res.allobs$response.name == "W")[,c(3,4,15:19,32:36,49,9:14)],
      subset(res.kfold, res.kfold$response.name == "W")[,c(10:15)])[ord[1:10],],
cbind(subset(res.allobs, res.allobs$response.name == "B")[,c(3,4,15:19,32:36,49,9:14)],
      subset(res.kfold, res.kfold$response.name == "B")[,c(10:15)])[ord[1:10],],
cbind(subset(res.allobs, res.allobs$response.name == "F")[,c(3,4,15:19,32:36,49,9:14)],
      subset(res.kfold, res.kfold$response.name == "F")[,c(10:15)])[ord[1:10],]
)
write.csv(all_resp_top10_rank, "top10rank.csv")

# create the pareto front figure shown in the main text of the manuscript:
library(scales)
res <- cbind(res.allobs[-c(15:48)], res.kfold[,10:15])
res$LOOCV_R2 <- res.LOOCV$Rsquared
res$Best5 <- res$covariates %in% res.allobs$covariates[ord[1:5]]
res$size <- c(0.8, 1)[res$Best5+1]
p1 <- ggplot(data = res, aes(x = MAPE, y = PRSEmax, color = Rsquared))+#color = LOOCV_R2)) +
  geom_point(data = subset(res, res$Rsquared > 0.8), aes(shape = Best5, size = Best5)) +
  scale_shape_manual(values=c(1, 6)) +
  scale_size_manual(values = c(1, 2)) +
  #scale_colour_gradient(low = grey(0.8), high = grey(0.2)) +
  scale_color_gradientn(colours = viridis_pal()(10)[3:8]) +
  #geom_point(data = subset(res, res$Rsquared < 0.9), aes(shape = Best5, size = Best5),
  #           color = viridis_pal()(10)[2]) +
  geom_point(data = subset(res, res$Rsquared < 0.8), aes(shape = Best5, size = Best5),
             color = viridis_pal()(10)[1]) +
  scale_x_log10() +
  scale_y_log10() +
  facet_wrap(response.name~.) +
  theme_bw()
p1
ggsave("pareto_front_with_MAPE_col.svg", width = 6.5, height = 5)

# create model diagnosis plots for the 5 best models:
pdf("model_diagnostic_plots_5best_scalesum_models.pdf")
for (j in 1:4){
  response.name <- c("TAGB", "W" , "B", "F")[j]
  for (i in 1:5){ # evalaute the 5 best models (according to ranksum statistics)
    response <- paste("log(",response.name,")",sep ="") # log transformation
    covariates <- covariate_functions[ord[i]] # evaluate each covariate function
    form <- as.formula(paste(response, " ~ ", covariates, sep = "")) # formula
    m.def <- lm(form, data = obs)
    par(mfrow=c(2,2));plot(m.def);title(main = form, outer = TRUE, line = -1)
  }
}
dev.off()

# create model predictions plots for the 5 best models:
pdf("model_predictions_5best_scalesum_models.pdf")
for (i in 1:5){ # make predictions with the 5 best models (according to ranksum statistics)
  par(mfrow=c(2,2), oma = c(0,0,2,0), pty = "s")
  for (j in 1:4){
    response.name <- c("TAGB", "W" , "B", "F")[j]
    response <- paste("log(",response.name,")",sep ="") # log transformation
    covariates <- covariate_functions[ord[i]] # evaluate each covariate function
    form <- as.formula(paste(response, " ~ ", covariates, sep = "")) # formula
    m.def <- lm(form, data = obs)
    plot(y = exp(m.def$fitted.values), x = obs[,response.name],
         ylab = paste("exp ( predicted",response.name,")"),
         xlab = paste("observed ",response.name, sep = ""),
         ylim = c(0,c(12500,6500,1750,4500)[j]))
    abline(a = 0, b = 1, lty = 3, lwd = 0.5)
    title(main = paste(c("Total Above Ground Biomass","Wood","Bark","Foliage")[j],"(kg)"))
    title(main = paste("RMSE (kg):",
                       round(subset(res.allobs$RMSEkg,
                                    res.allobs$response == response & res.allobs$covariates ==
covariates))),
          line = -0.75, cex.main = 0.75)
    title(main = paste("max. PRSE (%):",
                       round(subset(res.allobs$PRSEmax,
                                    res.allobs$response == response & res.allobs$covariates ==
covariates))),
          line = -1.75, cex.main = 0.75)
  }
  title(main = paste("Equation: log(response) ~",covariates), outer = TRUE, line = 0,
        font = 3)
}
dev.off()
```