Supplementary Materials

A R program for calculating NLME model assessment results and different subsample calibration results, using leave-one-site-out cross-validation.

```
setwd("E:/R/Remote sensing")
d<-read.csv('HCB.csv',header=T)
library(nlme)
library(Matrix)

ErrorStatistics<-function(Est,Obs){
   Index<-array(dim=4)
   Bias<-sum(Obs-Est)/length(Obs)
   EBias<-(sum(Obs-Est)/length(Obs))/mean(Obs)
   MAE<-abs(sum(Obs-Est))/length(Obs)
   EMAE<-(abs(sum(Obs-Est))/length(Obs))/mean(Obs)
   Index[1]<-Bias
   Index[2]<-EBias
   Index[3]<-MAE
   Index[4]<-EMAE
   dimnames(Index)<-list(c("Bias","EBias","MAE","EMAE"))
   return(Index)
}

Fun<-expression(H*(1-exp(b0+u01+u11+(b1+u02+u12)*CW+b2*CCp75+b3*H99)))
predictionResult = d
predictionResult$PredFixed = NaN
predictionResult$PredSiteLevel = NaN
predictionResult$PredSitePlotLevel = NaN

# leave-one-site-out cross-validation of NLME model
for (i in unique(d$Ref_Site)) {
    sampledata <-d[d$Ref_Site %in% i,]          #Extract the i-th site
    modeling_data <-d[!d$Ref_Site %in% i,]          #Extract the sites except the i-th site
    nlm2<-nlme(HCB~H*(1-exp(b0+b1*CW+b2*CCp75+b3*H99)),
                    fixed=b0+b1+b2+b3~1,
            random=list(Ref_Site=pdLogChol(b0+b1~1),Ref_PLOT=pdLogChol(b0+b1~1)),
                    start=c(b0=-0.1, b1=0.05, b2=-0.1,b3=-0.05),
                    data= modeling_data,
                    weights = varPower(form = ~H),
                    method = 'REML')

   # fixed parameters
    p<-nlm2$coefficients$fixed
```

```
b0<-p[1]
b1<-p[2]
b2<-p[3]
b3<-p[4]
u01 = u02 = u11 = u12 = 0

CW <- sampledata $CW
H    <- sampledata $H
CCp75 <- sampledata $CCp75
H99 <- sampledata $H99
HCB<- sampledata $HCB

predictionResult$PredFixed[predictionResult$Ref_Site==i] = eval(Fun) #Fixed prediction

# R calculation
qq <- nlm2$modelStruct$varStruct
ee<-as.matrix(HCB-eval(Fun))
temp<-H^(qq*2)
Gmatrix<-diag(temp)
R<-(nlm2$sigma^2)*sqrt(Gmatrix)%*%diag(rep(1,length(H)))%*%sqrt(Gmatrix)

# Z/G calculation by plot iteration
f11<-D(Fun,'u01')
f12<-D(Fun,'u02')
f21<-D(Fun,'u11')
f22<-D(Fun,'u12')

SiteZ = matrix(nrow = 0,ncol = 2)
PlotZ = matrix(nrow = 0, ncol = 0)

  # random-effect variance-covariance matrix
G2 <- as.matrix(cov(nlm2$coefficients$random$Ref_Site))
G1 <- as.matrix(cov(nlm2$coefficients$random$Ref_PLOT))

Site_i_AllPlot = unique(sampledata $Ref_PLOT)

for (j in Site_i_AllPlot){
    G = as.matrix(bdiag(G,G1))
    Site_i_Plot_j <- subset(sampledata, Ref_PLOT==j)
    CW <- Site_i_Plot_j $CW
    H    <- Site_i_Plot_j $H
    CCp75 <- Site_i_Plot_j$ CCp75
    H99 <- Site_i_Plot_j $H99
    HCB<- Site_i_Plot_j $HCB
```

```
        z11<-eval(f11)
        z12<-eval(f12)
        z21<-eval(f21)
        z22<-eval(f22)
        SiteZ = rbind(SiteZ,cbind(z11,z12))
        PlotZ = as.matrix(bdiag(PlotZ,cbind(z21,z22)))
    }

    Z = cbind(SiteZ,PlotZ)
    A <- Z%*%G%*%t(Z)+R
    u <- G%*%t(Z)%*%solve(A)%*%ee

    u01 <- u[1]
    u02 <- u[2]

    for (x in 1:length(Site_i_AllPlot)){
        u11 = u12 = 0
        Site_i_Plot_x <- subset(sampledata, Ref_PLOT==Site_i_AllPlot[x])

        CW <- Site_i_Plot_x $CW
        H    <- Site_i_Plot_x $H
        CCp75 <- Site_i_Plot_x $CCp75
        H99 <- Site_i_Plot_x $H99
        HCB<- Site_i_Plot_x $HCB

        predictionResult$PredSiteLevel[predictionResult$Ref_PLOT==Site_i_AllPlot[x]] =
            eval(Fun) #SiteLevel prediction

        u11 <- u[(x-1)*2+1+2]
        u12 <- u[(x-1)*2+2+2]

        predictionResult$PredSitePlotLevel[predictionResult$Ref_PLOT==Site_i_AllPlot[x]] =
            eval(Fun)         #SitePlotLevel prediction
    }
}

#Error Statistics
ErrorStatistics(predictionResult$PredSitePlotLevel, predictionResult$HCB)    # calibrated
ErrorStatistics(predictionResult$PredFixed, predictionResult$HCB) # un-calibrated


# Comparison of different sampling strategies (for example, sampling the largest tree (type II))
predictionResult1 = d
predictionResult1$PredSitePlotLevel = NaN
```

```
r<-matrix(nrow=18,ncol=4) #CalibrationResult
for (k in c(1:18)) {
   for (i in unique(d$Ref_Site)) {
     sampledata <-d[d$Ref_Site %in% i,]          #Extract the i-th site
     modeling_data <-d[!d$Ref_Site %in% i,]          #Extract the sites except the i-th site
     nlm2<-nlme(HCB~H*(1-exp(b0+b1*CW+b2*CCp75+b3*H99)),
                      fixed=b0+b1+b2+b3~1,
            random=list(Ref_Site=pdLogChol(b0+b1~1),Ref_PLOT=pdLogChol(b0+b1~1)),
                      start=c(b0=-0.1, b1=0.05, b2=-0.1,b3=-0.05),
                      data= modeling_data,
                      weights = varPower(form = ~H),
                      method = 'REML')

      subsampledata <-data.frame()
      for (j in unique(sampledata $Ref_PLOT)) {
            Plot_j = sampledata [sampledata $Ref_PLOT%in%j,]
            Plot_j1<-Plot_j[order(-Plot_j$DBH),]
            Plot_j1<-Plot_j1[1:k,]
            subsampledata = rbind(subsampledata, Plot_j1)
        }
      p<-nlm2$coefficients$fixed
      b0<-p[1]
      b1<-p[2]
      b2<-p[3]
      b3<-p[4]
      u01 = u02 = u11 = u12 = 0

      CW <- subsampledata $CW
      H    <- subsampledata $H
      CCp75 <- subsampledata $CCp75
      H99 <- subsampledata $H99
      HCB<- subsampledata $HCB

      # R calculation
      qq <- nlm2$modelStruct$varStruct
      ee<-as.matrix(HCB-eval(Fun))
      temp<-H^(qq*2)
      Gmatrix<-diag(temp)
      R<-(nlm2$sigma^2)*sqrt(Gmatrix)%*%diag(rep(1,length(H)))%*%sqrt(Gmatrix)

      # Z/G calculation by plot iteration
      f11<-D(Fun,'u01')
      f12<-D(Fun,'u02')
      f21<-D(Fun,'u11')
```

```
f22<-D(Fun,'u12')

SiteZ = matrix(nrow = 0,ncol = 2)
PlotZ = matrix(nrow = 0, ncol = 0)

G2 <- as.matrix(cov(nlm2$coefficients$random$Ref_Site))
G1 <- as.matrix(cov(nlm2$coefficients$random$Ref_PLOT))
G = G2

Site_i_AllPlot = unique(subsampledata $Ref_PLOT)

for (x in Site_i_AllPlot){

   G = as.matrix(bdiag(G,G1))

   Subsampledata_Plot_x <- subset(subsampledata, Ref_PLOT==x)

   CW <- Subsampledata_Plot_x $CW
   H    <- Subsampledata_Plot_x $H
   CCp75 <- Subsampledata_Plot_x $CCp75
   H99 <- Subsampledata_Plot_x $H99
   HCB<- Subsampledata_Plot_x $HCB
   z11<-eval(f11)
   z12<-eval(f12)
   z21<-eval(f21)
   z22<-eval(f22)

   SiteZ = rbind(SiteZ,cbind(z11,z12))
   PlotZ = as.matrix(bdiag(PlotZ,cbind(z21,z22)))
}
Z = cbind(SiteZ,PlotZ)
A <- Z%*%G%*%t(Z)+R
u <- G%*%t(Z)%*%solve(A)%*%ee

u01 <- u[1]
u02 <- u[2]

for (xx in 1:length(Site_i_AllPlot)){

   u11 = u12 = 0

   Site_i_Plot_xx <- subset(sampledata, Ref_PLOT==Site_i_AllPlot[xx])

   CW <- Site_i_Plot_xx $CW
```

```
        H    <- Site_i_Plot_xx $H
        CCp75 <- Site_i_Plot_xx $CCp75
        H99 <- Site_i_Plot_xx $H99
        HCB<- Site_i_Plot_xx $HCB
        u11 <- u[(xx-1)*2+1+2]
        u12 <- u[(xx-1)*2+2+2]
        predictionResult1$PredSitePlotLevel[predictionResult1$Ref_PLOT==Site_i_AllPlot[xx]]
= eval(Fun)
      }
   }
   r[k,1]<-ErrorStatistics(predictionResult1$PredSitePlotLevel,predictionResult$HCB)[1]
   r[k,2]<-ErrorStatistics(predictionResult1$PredSitePlotLevel,predictionResult$HCB)[2]
   r[k,3]<-ErrorStatistics(predictionResult1$PredSitePlotLevel,predictionResult$HCB)[3]
   r[k,4]<-ErrorStatistics(predictionResult1$PredSitePlotLevel,predictionResult$HCB)[4]
}
```