

# Spectra-trait PLSR example using leaf-level spectra and leaf mass per area (LMA) data from CONUS NEON sites

Shawn P. Serbin, Julien Lamour, & Jeremiah Anderson

## Overview

This is an R Markdown Notebook to illustrate how to retrieve a dataset from the EcoSIS spectral database, choose the “optimal” number of pls components, and fit a pls model for leaf-mass area (LMA)

## Getting Started

### Installation

```
## Loading required package: usethis

##
## Attaching package: 'remotes'

## The following objects are masked from 'package:devtools':
##
##   dev_package_deps, install_bioc, install_bitbucket, install_cran,
##   install_deps, install_dev, install_git, install_github,
##   install_gitlab, install_local, install_svn, install_url,
##   install_version, update_packages

## The following object is masked from 'package:usethis':
##
##   git_credentials

##
## Attaching package: 'pls'

## The following object is masked from 'package:stats':
##
##   loadings

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##   filter, lag

## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

## here() starts at /Users/sserbin/Data/GitHub/PLSR_for_plant_trait_prediction
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##      combine
```

## Setup other functions and options

```
### Setup other functions and options
github_dir <- file.path(here::here(), "R_Scripts")
source_from_gh <- TRUE
if (source_from_gh) {
  # Source helper functions from GitHub
  print("*** GitHub hash of functions.R file:")
  devtools::source_url("https://raw.githubusercontent.com/TESTgroup-BNL/PLSR_for_plant_trait_prediction")
} else {
  functions <- file.path(github_dir, "functions.R")
  source(functions)
}
```

```
## [1] "*** GitHub hash of functions.R file:"
## SHA-1 hash of file is d8af92cac6fa9c1337442724485b786577bc6259
```

```
# not in
`%notin%` <- Negate(`%in%`)

# Script options
pls::pls.options(plsralg = "oscorespls")
pls::pls.options("plsralg")
```

```
## $plsralg
## [1] "oscorespls"
```

```
# Default par options
opar <- par(no.readonly = T)

# What is the target variable?
inVar <- "LMA_gDW_m2"

# What is the source dataset from EcoSIS?
ecosis_id <- "5617da17-c925-49fb-b395-45a51291bd2d"
```

## Set working directory (scratch space)

```
## [1] "Output directory: /private/var/folders/xp/h3k9vf3n2jx181ts786_yjrn9c2gjq/T/Rtmpog0ykS"
```

## Grab data from EcoSIS

```
print(paste0("Output directory: ", getwd())) # check wd
```

URL: <https://ecosis.org/package/fresh-leaf-spectra-to-estimate-lma-over-neon-domains-in-eastern-united-states>

```
## [1] "Output directory: /Users/sserbin/Data/GitHub/PLSR_for_plant_trait_prediction/vignettes"
```

```
### Get source dataset from EcoSIS
```

```
dat_raw <- get_ecosis_data(ecosis_id = ecosis_id)
```

```
## [1] "**** Downloading Ecosis data ****"
```

```
## Downloading data...
```

```
## Parsed with column specification:
```

```
## cols(  
##   .default = col_double(),  
##   Affiliation = col_character(),  
##   `Common Name` = col_character(),  
##   Domain = col_character(),  
##   Functional_type = col_character(),  
##   `Latin Genus` = col_character(),  
##   `Latin Species` = col_character(),  
##   PI = col_character(),  
##   Project = col_character(),  
##   Sample_ID = col_character(),  
##   `USDA Symbol` = col_character()  
## )
```

```
## See spec(...) for full column specifications.
```

```
## Download complete!
```

```
head(dat_raw)
```

```
## # A tibble: 6 x 2,162  
##   Affiliation `Common Name` Domain Functional_type LMA `Latin Genus`  
##   <chr>      <chr>      <chr> <chr>          <dbl> <chr>  
## 1 University~ black walnut D02    broadleaf      72.9 Juglans  
## 2 University~ black walnut D02    broadleaf      72.9 Juglans  
## 3 University~ black walnut D02    broadleaf      60.8 Juglans  
## 4 University~ black walnut D02    broadleaf      60.8 Juglans  
## 5 University~ black walnut D02    broadleaf      85.9 Juglans  
## 6 University~ black walnut D02    broadleaf      85.9 Juglans  
## # ... with 2,156 more variables: `Latin Species` <chr>, PI <chr>,  
## #   Project <chr>, Sample_ID <chr>, `USDA Symbol` <chr>, `350` <dbl>,  
## #   `351` <dbl>, `352` <dbl>, `353` <dbl>, `354` <dbl>, `355` <dbl>,  
## #   `356` <dbl>, `357` <dbl>, `358` <dbl>, `359` <dbl>, `360` <dbl>,  
## #   `361` <dbl>, `362` <dbl>, `363` <dbl>, `364` <dbl>, `365` <dbl>,  
## #   `366` <dbl>, `367` <dbl>, `368` <dbl>, `369` <dbl>, `370` <dbl>,  
## #   `371` <dbl>, `372` <dbl>, `373` <dbl>, `374` <dbl>, `375` <dbl>,  
## #   `376` <dbl>, `377` <dbl>, `378` <dbl>, `379` <dbl>, `380` <dbl>,  
## #   `381` <dbl>, `382` <dbl>, `383` <dbl>, `384` <dbl>, `385` <dbl>,  
## #   `386` <dbl>, `387` <dbl>, `388` <dbl>, `389` <dbl>, `390` <dbl>,  
## #   `391` <dbl>, `392` <dbl>, `393` <dbl>, `394` <dbl>, `395` <dbl>,  
## #   `396` <dbl>, `397` <dbl>, `398` <dbl>, `399` <dbl>, `400` <dbl>,  
## #   `401` <dbl>, `402` <dbl>, `403` <dbl>, `404` <dbl>, `405` <dbl>,  
## #   `406` <dbl>, `407` <dbl>, `408` <dbl>, `409` <dbl>, `410` <dbl>,  
## #   `411` <dbl>, `412` <dbl>, `413` <dbl>, `414` <dbl>, `415` <dbl>,  
## #   `416` <dbl>, `417` <dbl>, `418` <dbl>, `419` <dbl>, `420` <dbl>,
```

```
## # `421` <dbl>, `422` <dbl>, `423` <dbl>, `424` <dbl>, `425` <dbl>,
## # `426` <dbl>, `427` <dbl>, `428` <dbl>, `429` <dbl>, `430` <dbl>,
## # `431` <dbl>, `432` <dbl>, `433` <dbl>, `434` <dbl>, `435` <dbl>,
## # `436` <dbl>, `437` <dbl>, `438` <dbl>, `439` <dbl>, `440` <dbl>,
## # `441` <dbl>, `442` <dbl>, `443` <dbl>, `444` <dbl>, ...
```

```
names(dat_raw)[1:40]
```

```
## [1] "Affiliation"      "Common Name"      "Domain"           "Functional_type"
## [5] "LMA"              "Latin Genus"      "Latin Species"    "PI"
## [9] "Project"          "Sample_ID"        "USDA Symbol"      "350"
## [13] "351"              "352"              "353"              "354"
## [17] "355"              "356"              "357"              "358"
## [21] "359"              "360"              "361"              "362"
## [25] "363"              "364"              "365"              "366"
## [29] "367"              "368"              "369"              "370"
## [33] "371"              "372"              "373"              "374"
## [37] "375"              "376"              "377"              "378"
```

## Create full pls dataset

```
### Create pls dataset
```

```
Start.wave <- 500
End.wave <- 2400
wv <- seq(Start.wave,End.wave,1)
Spectra <- as.matrix(dat_raw[,names(dat_raw) %in% wv])
colnames(Spectra) <- c(paste0("Wave_",wv))
sample_info <- dat_raw[,names(dat_raw) %notin% seq(350,2500,1)]
head(sample_info)
```

```
## # A tibble: 6 x 11
##   Affiliation `Common Name` Domain Functional_type LMA `Latin Genus`
##   <chr>      <chr>      <chr> <chr>      <dbl> <chr>
## 1 University~ black walnut D02    broadleaf    72.9 Juglans
## 2 University~ black walnut D02    broadleaf    72.9 Juglans
## 3 University~ black walnut D02    broadleaf    60.8 Juglans
## 4 University~ black walnut D02    broadleaf    60.8 Juglans
## 5 University~ black walnut D02    broadleaf    85.9 Juglans
## 6 University~ black walnut D02    broadleaf    85.9 Juglans
## # ... with 5 more variables: `Latin Species` <chr>, PI <chr>, Project <chr>,
## #   Sample_ID <chr>, `USDA Symbol` <chr>
```

```
sample_info2 <- sample_info %>%
  select(Domain,Functional_type,Sample_ID,USDA_Species_Code=`USDA Symbol`,LMA_gDW_m2=LMA)
head(sample_info2)
```

```
## # A tibble: 6 x 5
##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2
##   <chr> <chr>      <chr>      <chr>      <dbl>
## 1 D02    broadleaf    P0001    JUNI      72.9
## 2 D02    broadleaf    L0001    JUNI      72.9
## 3 D02    broadleaf    P0002    JUNI      60.8
## 4 D02    broadleaf    L0002    JUNI      60.8
## 5 D02    broadleaf    P0003    JUNI      85.9
## 6 D02    broadleaf    L0003    JUNI      85.9
```

```
plsr_data <- data.frame(sample_info2,Spectra)
rm(sample_info,sample_info2,Spectra)
```

## Create cal/val datasets

```
### Create cal/val datasets
## Make a stratified random sampling in the strata USDA_Species_Code and Domain

method <- "dplyr" #base/dplyr
# base R - a bit slow
# dplyr - much faster
split_data <- create_data_split(approach=method, split_seed=2356812, prop=0.8,
                                group_variables=c("USDA_Species_Code","Domain"))
names(split_data)
```

```
## [1] "cal_data" "val_data"
cal.plsr.data <- split_data$cal_data
head(cal.plsr.data)[1:8]
```

```
##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2 Wave_500
## 1   D08      broadleaf   L2644             ACBA      44.18 0.04170800
## 2   D08      broadleaf   L2646             ACBA      41.71 0.05067800
## 3   D08      broadleaf   L2645             ACBA      40.66 0.04701700
## 4   D08      broadleaf   P2639             ACBA      44.18 0.04125300
## 5   D03      broadleaf   P0614             ACFL      52.91 0.03895800
## 6   D03      broadleaf   L0609             ACFL      81.67 0.04186169
##   Wave_501 Wave_502
## 1 0.04208700 0.04283700
## 2 0.05087600 0.05153500
## 3 0.04718200 0.04766500
## 4 0.04150300 0.04247100
## 5 0.03915100 0.03956200
## 6 0.04217802 0.04258768
```

```
val.plsr.data <- split_data$val_data
head(val.plsr.data)[1:8]
```

```
##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2 Wave_500
## 4923   D08      broadleaf   P2462             <NA>      21.10 0.044964
## 4924   D08      broadleaf   L2462             SANI      100.72 0.068921
## 4925   D08      broadleaf   P2463             <NA>      29.59 0.036254
## 4926   D08      broadleaf   L2463             SANI      96.48 0.051810
## 4927   D08      broadleaf   P2464             <NA>      31.08 0.056587
## 4928   D08      broadleaf   L2464             SANI      61.40 0.037310
##   Wave_501 Wave_502
## 4923 0.045854 0.046911
## 4924 0.069633 0.070254
## 4925 0.036999 0.037671
## 4926 0.052113 0.052896
## 4927 0.057006 0.057734
## 4928 0.037223 0.037671
```

```
rm(split_data)

# Datasets:
print(paste("Cal observations: ",dim(cal.plsr.data)[1],sep=""))

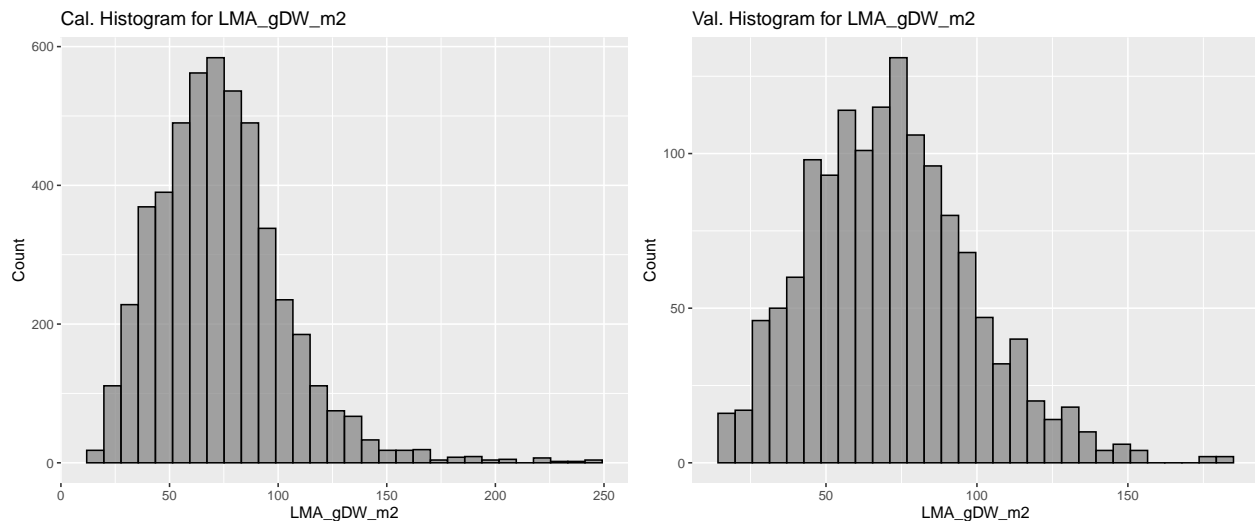
## [1] "Cal observations: 4922"

print(paste("Val observations: ",dim(val.plsr.data)[1],sep=""))

## [1] "Val observations: 1390"

cal_hist_plot <- qplot(cal.plsr.data[,paste0(inVar)],geom="histogram",
  main = paste0("Cal. Histogram for ",inVar),
  xlab = paste0(inVar),ylab = "Count",fill=I("grey50"),
  col=I("black"),alpha=I(.7))
val_hist_plot <- qplot(val.plsr.data[,paste0(inVar)],geom="histogram",
  main = paste0("Val. Histogram for ",inVar),
  xlab = paste0(inVar),ylab = "Count",fill=I("grey50"),
  col=I("black"),alpha=I(.7))
grid.arrange(cal_hist_plot, val_hist_plot, ncol=2)

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



## Create calibration and validation PLSR datasets

```
### Format PLSR data for model fitting
cal_spec <- as.matrix(cal.plsr.data[, which(names(cal.plsr.data) %in% paste0("Wave_",wv))])
cal.plsr.data <- data.frame(cal.plsr.data[, which(names(cal.plsr.data) %notin% paste0("Wave_",wv))],
  Spectra=I(cal_spec))
head(cal.plsr.data)[1:5]

##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2
## 1   D08      broadleaf    L2644             ACBA      44.18
## 2   D08      broadleaf    L2646             ACBA      41.71
## 3   D08      broadleaf    L2645             ACBA      40.66
## 4   D08      broadleaf    P2639             ACBA      44.18
```

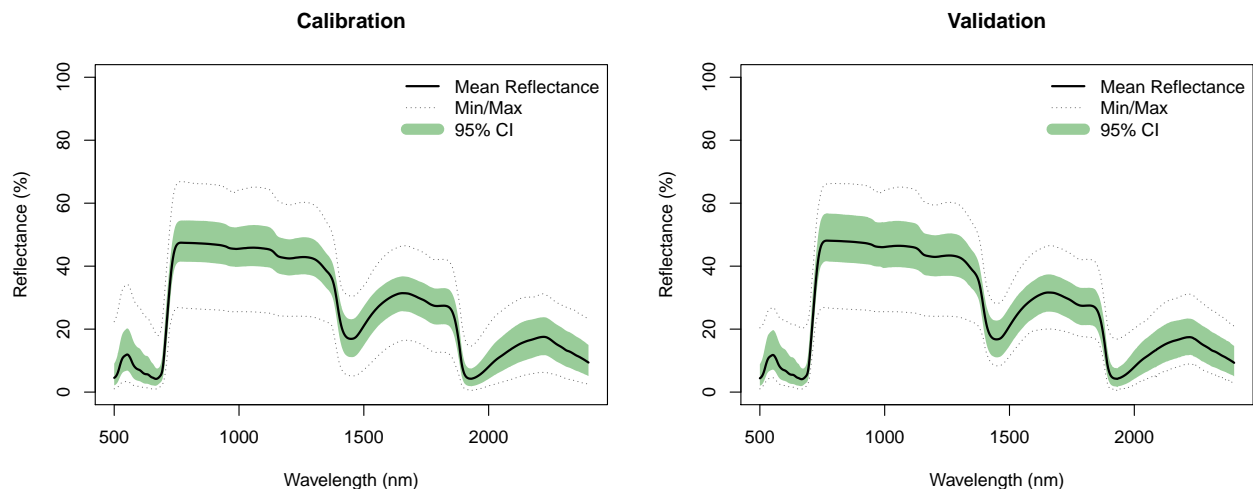
```
## 5      D03      broadleaf      P0614      ACFL      52.91
## 6      D03      broadleaf      L0609      ACFL      81.67
```

```
val_spec <- as.matrix(val.plsr.data[, which(names(val.plsr.data) %in% paste0("Wave_",wv))])
val.plsr.data <- data.frame(val.plsr.data[, which(names(val.plsr.data) %notin% paste0("Wave_",wv))],
                           Spectra=I(val_spec))
head(val.plsr.data)[1:5]
```

```
##      Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2
## 4923   D08      broadleaf      P2462      <NA>      21.10
## 4924   D08      broadleaf      L2462      SANI      100.72
## 4925   D08      broadleaf      P2463      <NA>      29.59
## 4926   D08      broadleaf      L2463      SANI      96.48
## 4927   D08      broadleaf      P2464      <NA>      31.08
## 4928   D08      broadleaf      L2464      SANI      61.40
```

plot cal and val spectra

```
par(mfrow=c(1,2)) # B, L, T, R
f.plot.spec(Z=cal.plsr.data$Spectra,wv=seq(Start.wave,End.wave,1),plot_label="Calibration")
f.plot.spec(Z=val.plsr.data$Spectra,wv=seq(Start.wave,End.wave,1),plot_label="Validation")
```



```
par(mfrow=c(1,1))
```

Use Jackknife permutation to determine optimal number of components

```
### Use permutation to determine the optimal number of components
if(grepl("Windows", sessionInfo()$running)){
  pls.options(parallel = NULL)
} else {
  pls.options(parallel = parallel::detectCores()-1)
}

method <- "custom" #pls, custom, lowestPRESS
random_seed <- 2356812
seg <- 100
maxComps <- 20
```

```

iterations <- 20
if (method=="pls") {
  # pls package approach - faster but estimates more components....
  nComps <- find_optimal_components(method=method, maxComps=maxComps, seg=seg,
                                    random_seed=random_seed)
  print(paste0("*** Optimal number of components: ", nComps))
} else {
  # custom method - slow but generally finds the smallest number of components
  nComps <- find_optimal_components(method=method, maxComps=maxComps, iterations=iterations,
                                    seg=seg, prop=0.70,
                                    random_seed=random_seed)
}

```

```
## [1] "*** Running jackknife permutation test. Please hang tight, this can take awhile ***"
```

```
## Running iteration 1
```

```
## Running iteration 2
```

```
## Running iteration 3
```

```
## Running iteration 4
```

```
## Running iteration 5
```

```
## Running iteration 6
```

```
## Running iteration 7
```

```
## Running iteration 8
```

```
## Running iteration 9
```

```
## Running iteration 10
```

```
## Running iteration 11
```

```
## Running iteration 12
```

```
## Running iteration 13
```

```
## Running iteration 14
```

```
## Running iteration 15
```

```
## Running iteration 16
```

```
## Running iteration 17
```

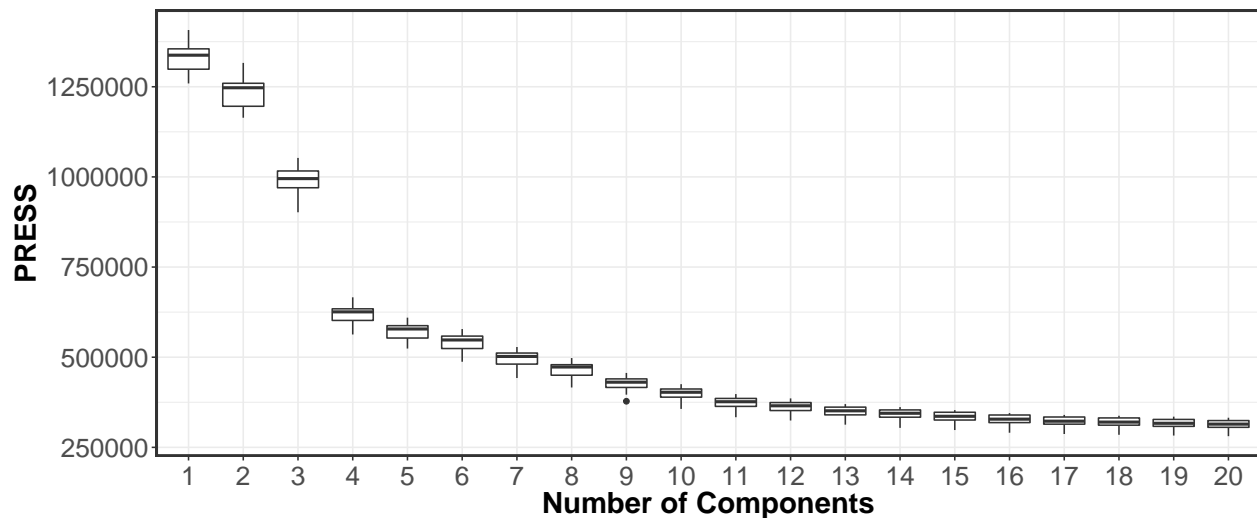
```
## Running iteration 18
```

```
## Running iteration 19
```

```
## Running iteration 20
```

```
## No id variables; using all as measure variables
```





```
## [1] "*** Optimal number of components based on t.test: 14"
```

Fit final model

```
### Fit final model
segs <- 100
plsr.out <- plsr(as.formula(paste(inVar,"~","Spectra")),scale=FALSE,ncomp=nComps,
                validation="CV",
                segments=segs, segment.type="interleaved",trace=FALSE,
                data=cal.plsr.data)
fit <- plsr.out$fitted.values[,1,nComps]
pls.options(parallel = NULL)
```

```
# External validation fit stats
par(mfrow=c(1,2)) # B, L, T, R
RMSEP(plsr.out, newdata = val.plsr.data)
```

```
## (Intercept)      1 comps      2 comps      3 comps      4 comps      5 comps
##      27.155      17.610      16.595      15.483      13.235      12.374
##      6 comps      7 comps      8 comps      9 comps     10 comps     11 comps
##      11.499      10.722      10.269      9.647       9.197       9.319
##     12 comps     13 comps     14 comps
##      9.515       9.403       9.143
```

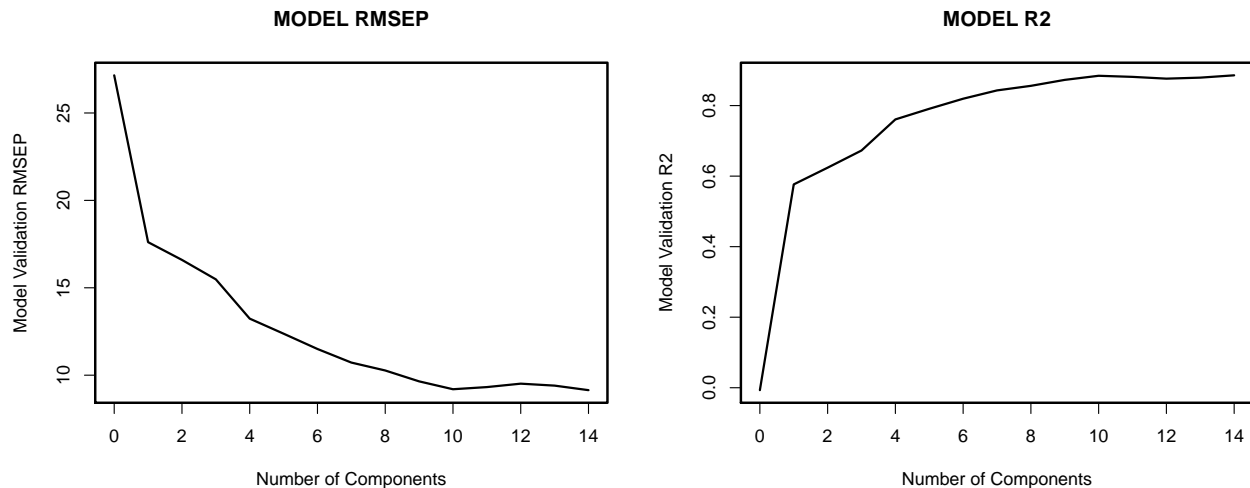
```
plot(RMSEP(plsr.out,estimate=c("test"),newdata = val.plsr.data),
     main="MODEL RMSEP",
     xlab="Number of Components",ylab="Model Validation RMSEP",lty=1,col="black",
     cex=1.5,lwd=2)
box(lwd=2.2)
```

```
R2(plsr.out, newdata = val.plsr.data)
```

```
## (Intercept)      1 comps      2 comps      3 comps      4 comps      5 comps
##    -0.006901     0.576543     0.623949     0.672643     0.760799     0.790906
##      6 comps      7 comps      8 comps      9 comps     10 comps     11 comps
##      0.819456     0.843031     0.856001     0.872913     0.884511     0.881406
##     12 comps     13 comps     14 comps
```

```
##      0.876368      0.879269      0.885848
```

```
plot(R2(plsr.out,estimate=c("test"),newdata = val.plsr.data), main="MODEL R2",
     xlab="Number of Components",ylab="Model Validation R2",lty=1,col="black",
     cex=1.5,lwd=2)
box(lwd=2.2)
```



```
par(opar)
```

### PLSR fit observed vs. predicted plot data

```
#calibration
cal.plsr.output <- data.frame(cal.plsr.data[, which(names(cal.plsr.data) %notin% "Spectra")],
                             PLSR_Predicted=fit,
                             PLSR_CV_Predicted=as.vector(plsr.out$validation$pred[,nComps]))
cal.plsr.output <- cal.plsr.output %>%
  mutate(PLSR_CV_Residuals = PLSR_CV_Predicted-get(inVar))
head(cal.plsr.output)
```

```
##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2 PLSR_Predicted
## 1   D08      broadleaf   L2644          ACBA      44.18      52.25180
## 2   D08      broadleaf   L2646          ACBA      41.71      40.47077
## 3   D08      broadleaf   L2645          ACBA      40.66      44.14454
## 4   D08      broadleaf   P2639          ACBA      44.18      44.96530
## 5   D03      broadleaf   P0614          ACFL      52.91      60.96168
## 6   D03      broadleaf   L0609          ACFL      81.67      91.11058
##   PLSR_CV_Predicted PLSR_CV_Residuals
## 1          52.32370          8.1436968
## 2          40.51303         -1.1969740
## 3          44.34736          3.6873566
## 4          45.12281          0.9428088
## 5          61.15931          8.2493103
## 6          91.19997          9.5299655
```

```
cal.R2 <- round(pls::R2(plsr.out)[[1]][nComps],2)
cal.RMSEP <- round(sqrt(mean(cal.plsr.output$PLSR_CV_Residuals^2)),2)

val.plsr.output <- data.frame(val.plsr.data[, which(names(val.plsr.data) %notin% "Spectra")],
```

```

                                PLSR_Predicted=as.vector(predict(plsr.out,
                                                                newdata = val.plsr.data,
                                                                ncomp=nComps, type="response")[,1]))
val.plsr.output <- val.plsr.output %>%
  mutate(PLSR_Residuals = PLSR_Predicted-get(inVar))
head(val.plsr.output)

##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2 PLSR_Predicted
## 1   D08      broadleaf   P2462             <NA>      21.10      23.17688
## 2   D08      broadleaf   L2462             SANI      100.72     87.76061
## 3   D08      broadleaf   P2463             <NA>      29.59     30.12776
## 4   D08      broadleaf   L2463             SANI      96.48     89.17843
## 5   D08      broadleaf   P2464             <NA>      31.08     45.66874
## 6   D08      broadleaf   L2464             SANI      61.40     65.10542
##   PLSR_Residuals
## 1      2.0768781
## 2     -12.9593944
## 3      0.5377642
## 4     -7.3015651
## 5     14.5887365
## 6      3.7054219

val.R2 <- round(pls::R2(plsr.out,newdata=val.plsr.data)[[1]][nComps],2)
val.RMSEP <- round(sqrt(mean(val.plsr.output$PLSR_Residuals^2)),2)

rng_quant <- quantile(cal.plsr.output[,inVar], probs = c(0.001, 0.999))
cal_scatter_plot <- ggplot(cal.plsr.output, aes(x=PLSR_CV_Predicted,
                                                y=get(inVar))) +
  theme_bw() + geom_point() + geom_abline(intercept = 0, slope = 1,
                                          color="dark grey",
                                          linetype="dashed",
                                          size=1.5) +
  xlim(rng_quant[1], rng_quant[2]) +
  ylim(rng_quant[1], rng_quant[2]) +
  labs(x=paste0("Predicted ", paste(inVar), " (units)"),
       y=paste0("Observed ", paste(inVar), " (units)"),
       title=paste0("Calibration: ", paste0("Rsq = ", cal.R2), "; ",
                paste0("RMSEP = ",
                cal.RMSEP))) +
  theme(axis.text=element_text(size=18), legend.position="none",
        axis.title=element_text(size=20, face="bold"),
        axis.text.x = element_text(angle = 0,vjust = 0.5),
        panel.border = element_rect(linetype = "solid",
                                     fill = NA, size=1.5))

cal_resid_histogram <- ggplot(cal.plsr.output,
                              aes(x=PLSR_CV_Residuals)) +
  geom_histogram(alpha=.5, position="identity") +
  geom_vline(xintercept = 0, color="black",
            linetype="dashed", size=1) + theme_bw() +
  theme(axis.text=element_text(size=18), legend.position="none",
        axis.title=element_text(size=20, face="bold"),
        axis.text.x = element_text(angle = 0,vjust = 0.5),
        panel.border = element_rect(linetype = "solid",
                                     fill = NA, size=1.5))

```

```

                                fill = NA, size=1.5))

rng_quant <- quantile(val.plsr.output[,inVar],
                      probs = c(0.001, 0.999))
val_scatter_plot <- ggplot(val.plsr.output,
                           aes(x=PLSR_Predicted, y=get(inVar))) +
  theme_bw() + geom_point() +
  geom_abline(intercept = 0, slope = 1, color="dark grey",
              linetype="dashed", size=1.5) +
  xlim(rng_quant[1], rng_quant[2]) +
  ylim(rng_quant[1], rng_quant[2]) +
  labs(x=paste0("Predicted ", paste(inVar), " (units)"),
       y=paste0("Observed ", paste(inVar), " (units)"),
       title=paste0("Validation: ", paste0("Rsqr = ", val.R2), "; ",
               paste0("RMSEP = ",
                       val.RMSEP))) +
  theme(axis.text=element_text(size=18), legend.position="none",
        axis.title=element_text(size=20, face="bold"),
        axis.text.x = element_text(angle = 0,vjust = 0.5),
        panel.border = element_rect(linetype = "solid", fill = NA,
                                     size=1.5))

val_resid_histogram <- ggplot(val.plsr.output, aes(x=PLSR_Residuals)) +
  geom_histogram(alpha=.5, position="identity") +
  geom_vline(xintercept = 0, color="black",
             linetype="dashed", size=1) + theme_bw() +
  theme(axis.text=element_text(size=18), legend.position="none",
        axis.title=element_text(size=20, face="bold"),
        axis.text.x = element_text(angle = 0,vjust = 0.5),
        panel.border = element_rect(linetype = "solid", fill = NA,
                                     size=1.5))

# plot cal/val side-by-side
grid.arrange(cal_scatter_plot, val_scatter_plot, cal_resid_histogram,
              val_resid_histogram,
              nrow=2,ncol=2)

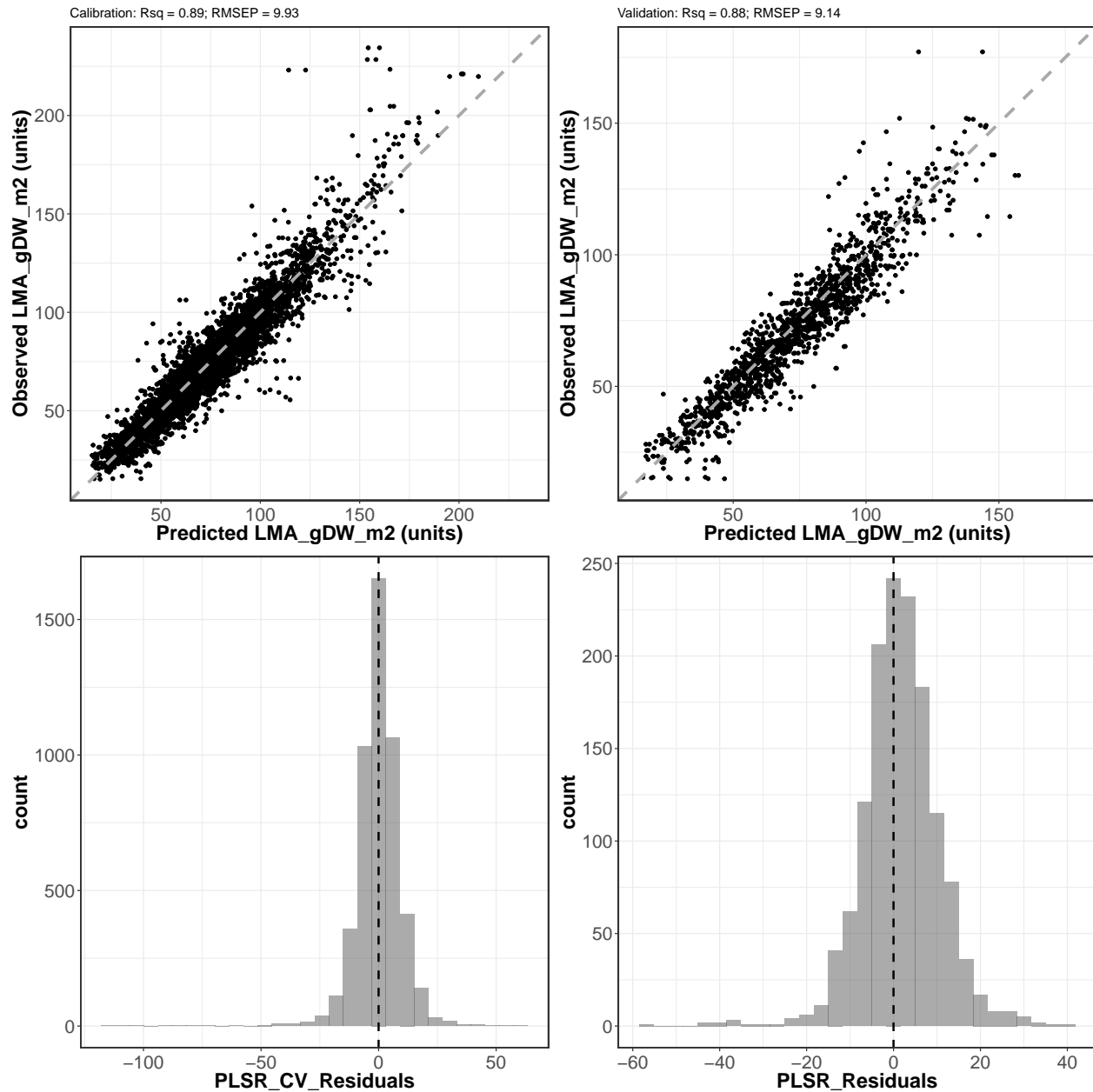
```

```
## Warning: Removed 16 rows containing missing values (geom_point).
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

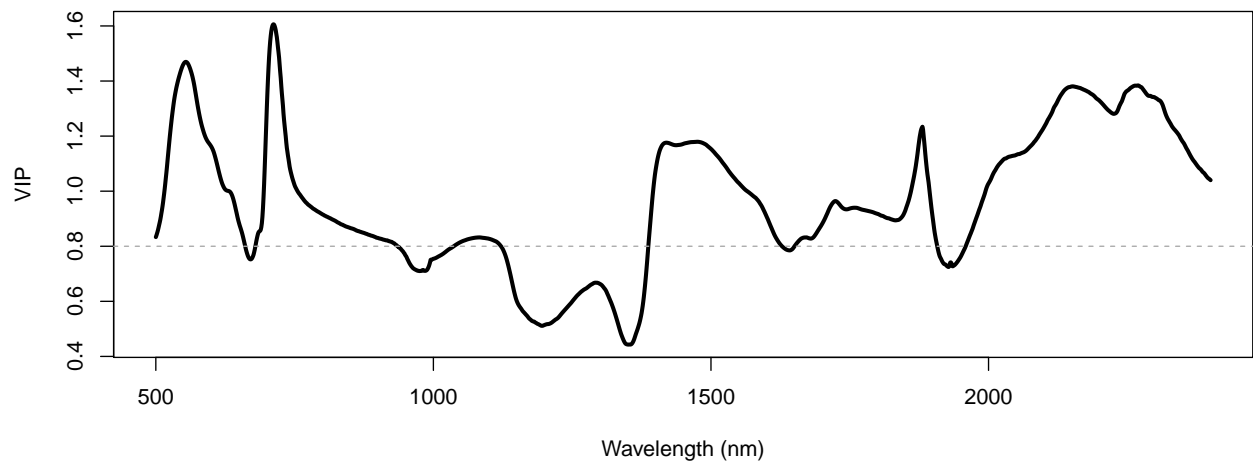
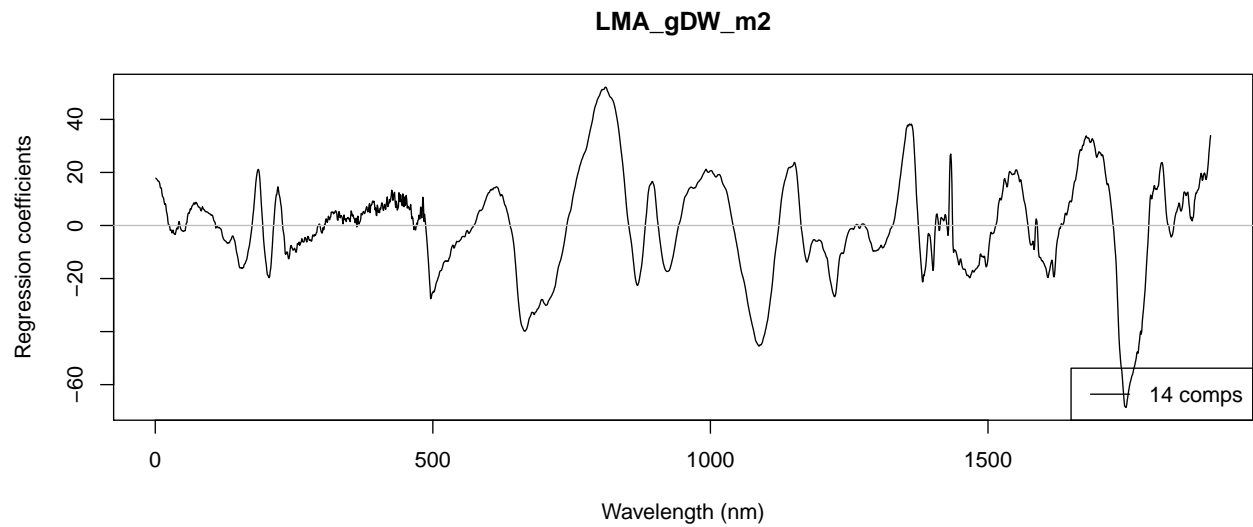
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Generate Coefficient and VIP plots

```
vips <- VIP(plsr.out)[nComps,]
par(mfrow=c(2,1))
plot(plsr.out, plottype = "coef", xlab="Wavelength (nm)",
     ylab="Regression coefficients", legendpos = "bottomright",
     ncomp=nComps)

plot(seq(Start.wave, End.wave, 1), vips, xlab="Wavelength (nm)", ylab="VIP",
     cex=0.01)
lines(seq(Start.wave, End.wave, 1), vips, lwd=3)
abline(h=0.8, lty=2, col="dark grey")
```



### Jackknife validation

```
if(grepl("Windows", sessionInfo()$running)){
  pls.options(parallel = NULL)
} else {
  pls.options(parallel = parallel::detectCores()-1)
}

seg <- 100
jk.plsr.out <- pls::plsr(as.formula(paste(inVar, "~", "Spectra")),
  scale=FALSE,
  center=TRUE, ncomp=nComps,
  validation="CV", segments = seg,
  segment.type="interleaved",
  trace=FALSE,
  jackknife=TRUE, data=cal.plsr.data)
pls.options(parallel = NULL)
```

```

Jackknife_coef <- f.coef.valid(plsr.out = jk.plsr.out,
                             data_plsr = cal.plsr.data,
                             ncomp = nComps)
Jackknife_intercept <- Jackknife_coef[1,,]
Jackknife_coef <- Jackknife_coef[2:dim(Jackknife_coef)[1],,,]

#interval <- c(0.025,0.975)
interval <- c(0.05,0.95)
Jackknife_Pred <- val.plsr.data$Spectra%*%Jackknife_coef+Jackknife_intercept
Interval_Conf <- apply(X = Jackknife_Pred,MARGIN = 1,
                      FUN = quantile,probs=c(interval[1],interval[2]))
Interval_Pred <- apply(X = Jackknife_Pred,MARGIN = 1,
                      FUN = quantile,probs=c(interval[1],interval[2]))
sd_mean <- apply(X = Jackknife_Pred,MARGIN = 1,FUN =sd)
sd_res <- sd(val.plsr.output$PLSR_Residuals)
sd_tot <- sqrt(sd_mean^2+sd_res^2)
val.plsr.output$LCI <- Interval_Pred[1,]
val.plsr.output$UCI <- Interval_Pred[2,]
val.plsr.output$LPI <- val.plsr.output$PLSR_Predicted+1.96*sd_tot
val.plsr.output$UPI <- val.plsr.output$PLSR_Predicted-1.96*sd_tot
head(val.plsr.output)

##   Domain Functional_type Sample_ID USDA_Species_Code LMA_gDW_m2 PLSR_Predicted
## 1    D08      broadleaf   P2462             <NA>      21.10      23.17688
## 2    D08      broadleaf   L2462             SANI      100.72     87.76061
## 3    D08      broadleaf   P2463             <NA>      29.59      30.12776
## 4    D08      broadleaf   L2463             SANI      96.48      89.17843
## 5    D08      broadleaf   P2464             <NA>      31.08      45.66874
## 6    D08      broadleaf   L2464             SANI      61.40      65.10542
##   PLSR_Residuals      LCI      UCI      LPI      UPI
## 1      2.0768781 22.71494 23.87684 40.82572 5.528041
## 2     -12.9593944 86.99439 88.32899 105.40986 70.111352
## 3      0.5377642 29.10105 31.53336 47.81203 12.443494
## 4     -7.3015651 88.39579 89.52677 106.82441 71.532455
## 5     14.5887365 45.01835 46.24715 63.31762 28.019855
## 6      3.7054219 64.37264 65.64146 82.75183 47.459017

```

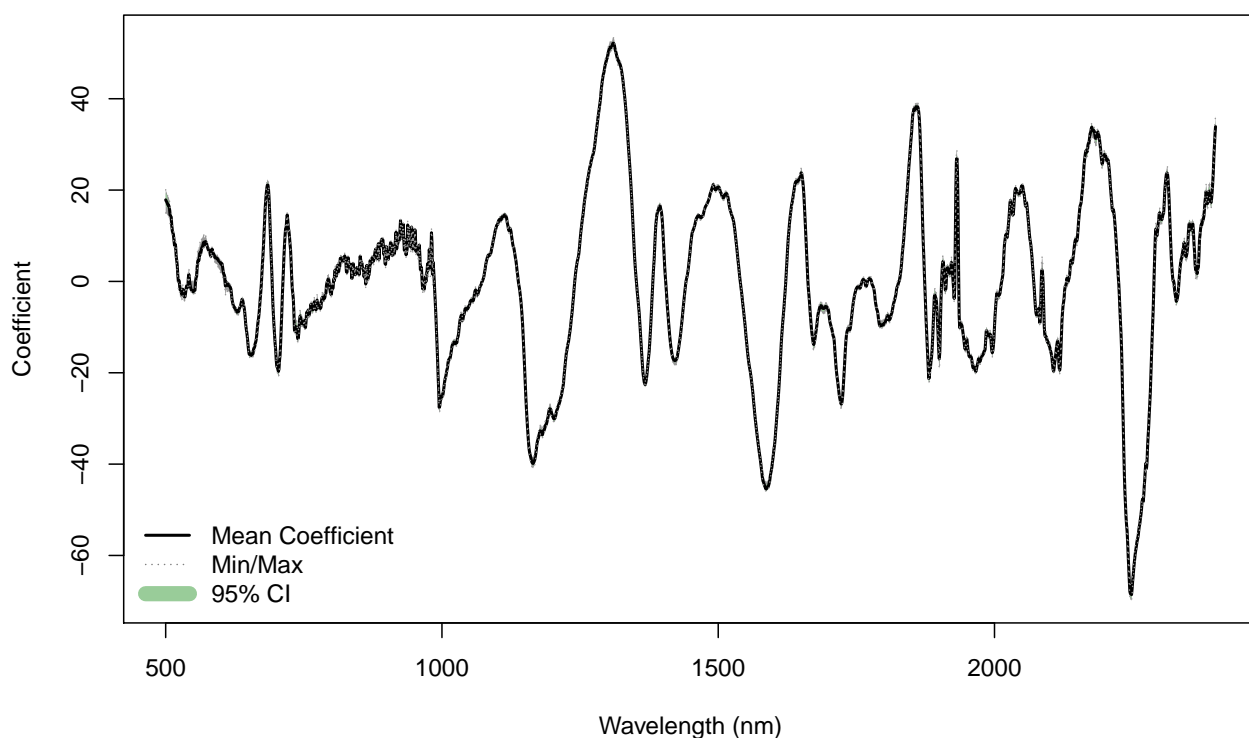
### Jackknife coefficient plot

```

f.plot.coef(Z = t(Jackknife_coef), wv = seq(Start.wave,End.wave,1),
            plot_label="Jackknife regression coefficients",
            position = 'bottomleft')

```

## Jackknife regression coefficients

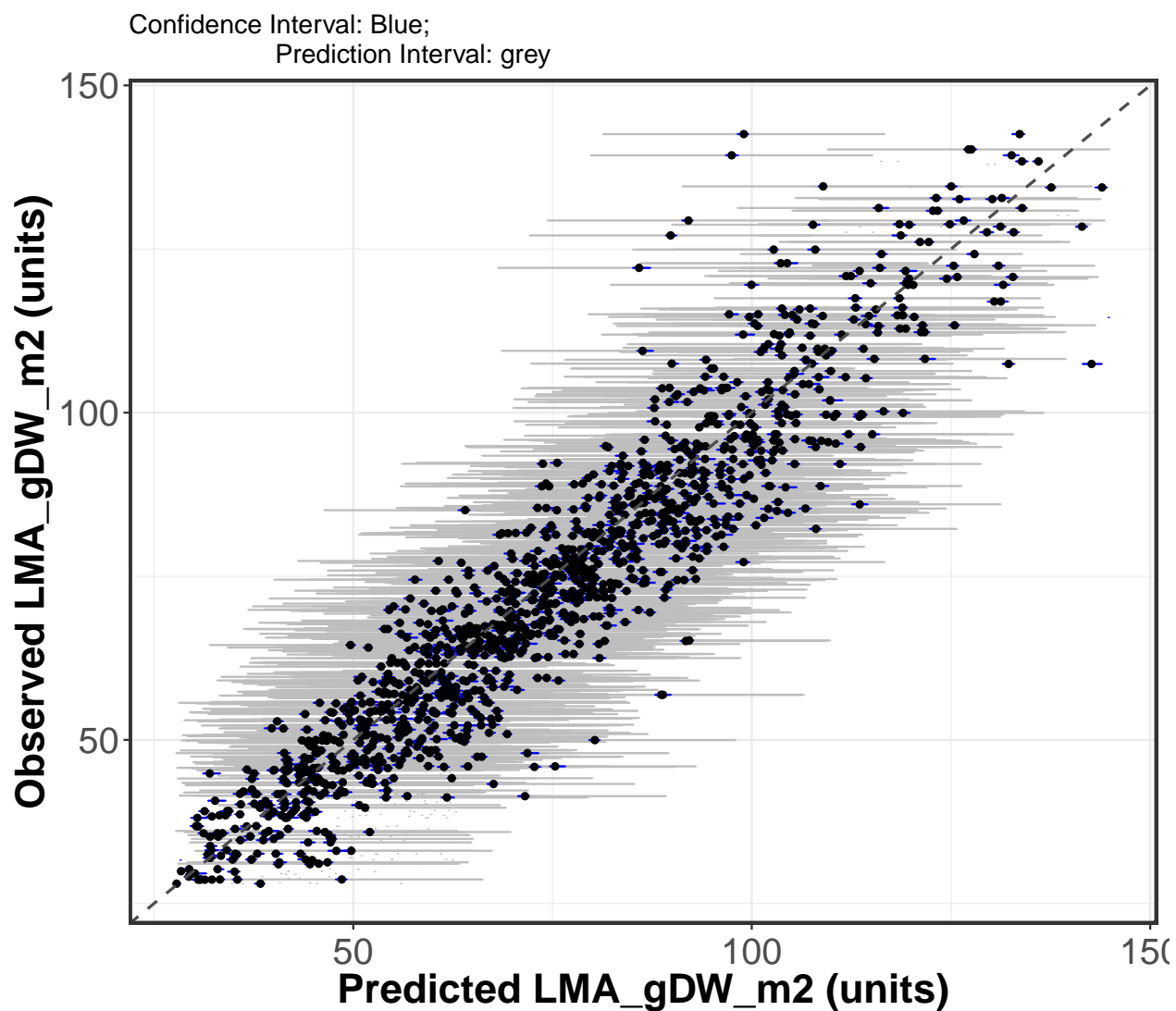


## Jackknife validation plot

```
rng_vals <- c(min(val.plsr.output$LPI), max(val.plsr.output$UPI))
jk_val_scatterplot <- ggplot(val.plsr.output, aes(x=PLSR_Predicted,
                                                  y=get(inVar))) +
  theme_bw() + geom_errorbar(aes(xmin = LPI, xmax = UPI), color='grey',
                              width=0.2) +
  geom_errorbar(aes(xmin = LCI, xmax = UCI), color='blue', width=0.2) +
  geom_point(size=1.3) +
  geom_abline(intercept = 0, slope = 1, color="grey30",
              linetype="dashed", size=0.7) +
  xlim(rng_vals[1], rng_vals[2]) +
  ylim(rng_vals[1], rng_vals[2]) +
  labs(x=paste0("Predicted ", paste(inVar), " (units)"),
       y=paste0("Observed ", paste(inVar), " (units)"),
       title=paste0("Confidence Interval: Blue;
                     Prediction Interval: grey")) +
  theme(axis.text=element_text(size=18),
        legend.position = 'right',
        axis.title=element_text(size=20, face="bold"),
        axis.text.x = element_text(angle = 0, vjust = 0.5),
        panel.border = element_rect(linetype = "solid",
                                     fill = NA, size=1.5))
print(jk_val_scatterplot)
```

## Warning: Removed 84 rows containing missing values (geom\_point).





Output jackknife results

```
out.jk.coefs <- data.frame(Iteration=seq(1,seg,1),
                           Intercept=Jackknife_intercept,
                           t(Jackknife_coef))
head(out.jk.coefs)[1:6]
```

```
##      Iteration Intercept Wave_500 Wave_501 Wave_502 Wave_503
## Seg 1         1  63.74332  18.24687  17.95401  17.77171  17.38132
## Seg 2         2  63.73713  18.85761  18.62012  18.39545  18.02299
## Seg 3         3  62.92261  19.42642  19.12960  18.88917  18.38110
## Seg 4         4  63.38407  18.33061  17.97824  17.88824  17.56360
## Seg 5         5  63.44702  17.60085  17.39623  17.12976  16.75319
## Seg 6         6  62.81849  17.70992  17.51663  17.39929  16.95014
```

```
write.csv(out.jk.coefs,file=file.path(outdir,
                                       paste0(inVar,
                                                '_Jackknife_PLSR_Coefficients.csv')),
```

```
row.names=FALSE)
```

## Create core PLSR outputs

```
print(paste("Output directory: ", getwd()))

## [1] "Output directory:  /Users/sserbin/Data/GitHub/PLSR_for_plant_trait_prediction/vignettes"
# Observed versus predicted
write.csv(cal.plsr.output,file=file.path(outdir,
                                         paste0(inVar,'_Observed_PLSR_CV_Pred_',
                                                  nComps,'comp.csv')),
          row.names=FALSE)

# Validation data
write.csv(val.plsr.output,file=file.path(outdir,
                                         paste0(inVar,'_Validation_PLSR_Pred_',
                                                  nComps,'comp.csv')),
          row.names=FALSE)

# Model coefficients
coefs <- coef(plsr.out,ncomp=nComps,intercept=TRUE)
write.csv(coefs,file=file.path(outdir,
                               paste0(inVar,'_PLSR_Coefficients_',
                                       nComps,'comp.csv')),
          row.names=TRUE)

# PLSR VIP
write.csv(vips,file=file.path(outdir,
                              paste0(inVar,'_PLSR_VIPs_',
                                      nComps,'comp.csv')))
```

## Confirm files were written to temp space

```
print("**** PLSR output files: ")

## [1] "**** PLSR output files: "
list.files(outdir)[grep(pattern = inVar, list.files(outdir))]

## [1] "LMA_gDW_m2_Jackkife_PLSR_Coefficients.csv"
## [2] "LMA_gDW_m2_Observed_PLSR_CV_Pred_14comp.csv"
## [3] "LMA_gDW_m2_PLSR_Coefficients_14comp.csv"
## [4] "LMA_gDW_m2_PLSR_VIPs_14comp.csv"
## [5] "LMA_gDW_m2_Validation_PLSR_Pred_14comp.csv"
```