Spectra-trait PLSR example using leaf-level spectra and leaf nitrogen content (Narea, g/m2) data from 36 species growing in Rosa rugosa invaded coastal grassland communities in Belgium

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 plsr components, and fit a plsr model for leaf nitrogen content (Narea, g/m2)

Getting Started

Installation

```
### Install and load required R packages
list.of.packages <- c("devtools","readr","RCurl","httr","pls","dplyr","reshape2","here",
                      "ggplot2","gridExtra")  # packages needed for script
# check for dependencies and install if needed
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)

# Load libraries
invisible(lapply(list.of.packages, library, character.only = TRUE))
```

```
## Loading required package: usethis

##
## 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 <- FALSE
if (source_from_gh) {
  # Source helper functions from GitHub
  devtools::source_url("https://raw.githubusercontent.com/TESTgroup-BNL/PLSR_for_plant_trait_prediction
} else {
  functions <- file.path(github_dir,"functions.R")
  source(functions)
}

# 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 <- "Narea_g_m2"

# What is the source dataset from EcoSIS?
ecosis_id <- "9db4c5a2-7eac-4e1e-8859-009233648e89"
```

Set working directory (scratch space)

```
## [1] "Output directory: /private/var/folders/xp/h3k9vf3n2jx181ts786_yjrn9c2gjq/T/RtmpfdW3fk"
```

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(),
##   `Latin Species` = col_character(),
##   ids = col_character(),
##   `plot code` = col_character(),
##   `species code` = col_character()
## )

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

## Download complete!
```

```
head(dat_raw)
```

```
## # A tibble: 6 x 2,164
##   `Cw/EWT (cm3/cm~ `Latin Species` `Leaf area (mm2~ `Leaf calcium c~
##              <dbl> <chr>                      <dbl>            <dbl>
## 1          0.00887 Arrhenatherum ~              696.           0.0291
## 2          0.00824 Bromus sterilis              447.           0.0230
## 3          0.0280  Jacobaea vulga~             2418.           0.0950
## 4          0.0106  Rubus caesius               5719.           0.0700
## 5          0.00851 Arrhenatherum ~              671.           0.0286
## 6          0.0153  Crepis capilla~             1401.           0.0470
## # ... with 2,160 more variables: `Leaf magnesium content per leaf area
## #   (mg/mm2)` <dbl>, `Leaf mass per area (g/cm2)` <dbl>, `Leaf nitrogen content
## #   per leaf area (mg/mm2)` <dbl>, `Leaf phosphorus content per leaf area
## #   (mg/mm2)` <dbl>, `Leaf potassium content per leaf area (mg/mm2)` <dbl>,
## #   `Plant height vegetative (cm)` <dbl>, ids <chr>, `plot code` <chr>,
## #   `species code` <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>, ...
```

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

```
##  [1] "Cw/EWT (cm3/cm2)"
##  [2] "Latin Species"
##  [3] "Leaf area (mm2)"
##  [4] "Leaf calcium content per leaf area (mg/mm2)"
##  [5] "Leaf magnesium content per leaf area (mg/mm2)"
##  [6] "Leaf mass per area (g/cm2)"
##  [7] "Leaf nitrogen content per leaf area (mg/mm2)"
##  [8] "Leaf phosphorus content per leaf area (mg/mm2)"
##  [9] "Leaf potassium content per leaf area (mg/mm2)"
## [10] "Plant height vegetative (cm)"
## [11] "ids"
## [12] "plot code"
## [13] "species code"
## [14] "350"
## [15] "351"
## [16] "352"
## [17] "353"
## [18] "354"
## [19] "355"
## [20] "356"
## [21] "357"
## [22] "358"
## [23] "359"
## [24] "360"
## [25] "361"
## [26] "362"
## [27] "363"
## [28] "364"
## [29] "365"
## [30] "366"
## [31] "367"
## [32] "368"
## [33] "369"
## [34] "370"
## [35] "371"
## [36] "372"
## [37] "373"
## [38] "374"
## [39] "375"
## [40] "376"
```

Create full plsr dataset

```
### Create plsr 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 13
##   `Cw/EWT (cm3/cm~ `Latin Species` `Leaf area (mm2~ `Leaf calcium c~
##              <dbl> <chr>                      <dbl>            <dbl>
## 1          0.00887 Arrhenatherum ~             696.           0.0291
## 2          0.00824 Bromus sterilis             447.           0.0230
## 3          0.0280  Jacobaea vulga~            2418.           0.0950
## 4          0.0106  Rubus caesius              5719.           0.0700
## 5          0.00851 Arrhenatherum ~             671.           0.0286
## 6          0.0153  Crepis capilla~            1401.           0.0470
## # ... with 9 more variables: `Leaf magnesium content per leaf area
## #   (mg/mm2)` <dbl>, `Leaf mass per area (g/cm2)` <dbl>, `Leaf nitrogen content
## #   per leaf area (mg/mm2)` <dbl>, `Leaf phosphorus content per leaf area
## #   (mg/mm2)` <dbl>, `Leaf potassium content per leaf area (mg/mm2)` <dbl>,
## #   `Plant height vegetative (cm)` <dbl>, ids <chr>, `plot code` <chr>,
## #   `species code` <chr>
```

```
sample_info2 <- sample_info %>%
  select(Plant_Species=`Latin Species`,Species_Code=`species code`,Plot=`plot code`,
         Narea_mg_mm2=`Leaf nitrogen content per leaf area (mg/mm2)`)
sample_info2 <- sample_info2 %>%
#  mutate(Narea_g_m2=Narea_mg_mm2*(0.001/1e-6)) # based on orig units should be this but conversion wro
  mutate(Narea_g_m2=Narea_mg_mm2*100) # this assumes orig units were g/mm2 or mg/cm2
head(sample_info2)
```

```
## # A tibble: 6 x 5
##   Plant_Species         Species_Code Plot  Narea_mg_mm2 Narea_g_m2
##   <chr>                 <chr>        <chr>        <dbl>      <dbl>
## 1 Arrhenatherum elatius Arrela       DC1        0.0126       1.26
## 2 Bromus sterilis       Broste       DC1        0.00682      0.682
## 3 Jacobaea vulgaris     Jacvul       DC1        0.0102       1.02
## 4 Rubus caesius         Rubcae       DC1        0.0121       1.21
## 5 Arrhenatherum elatius Arrela       DC2        0.0117       1.17
## 6 Crepis capillaris     Creves       DC2        0.00877      0.877
```

```
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=1245565, prop=0.8,
                                group_variables="Species_Code")
names(split_data)
```

```
## [1] "cal_data" "val_data"
```

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

```
##        Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2 Wave_500
## 1 Ammophila arenaria       Ammare  ZC3   0.03240495   3.240495 0.130885
## 2 Ammophila arenaria       Ammare  MC2   0.02806279   2.806279 0.135785
## 3 Ammophila arenaria       Ammare  ZC1   0.02041612   2.041612 0.147665
## 4 Ammophila arenaria       Ammare  MC1   0.02426549   2.426549 0.142765
## 5 Ammophila arenaria       Ammare  WC3   0.02807281   2.807281 0.151750
## 6 Ammophila arenaria       Ammare  WR3   0.02286678   2.286678 0.150850
##   Wave_501 Wave_502
## 1  0.13175 0.132750
## 2  0.13685 0.138150
## 3  0.14910 0.150330
## 4  0.14390 0.145200
## 5  0.15275 0.154150
## 6  0.15185 0.152815
```

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

```
##          Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2   Wave_500
## 184   Jacobaea vulgaris       Jacvul  WC2  0.008756996  0.8756996 0.06736887
## 185 Potentilla reptans       Potrep  WC2  0.010313464  1.0313464 0.07125000
## 186      Rubus caesius       Rubcae  WC2  0.007968454  0.7968454 0.05993560
## 187      Urtica dioica       Urtdio  WC2  0.012737560  1.2737560 0.06508300
## 188 Ammophila arenaria       Ammare  WC3  0.028072806  2.8072806 0.15175000
## 189   Jacobaea vulgaris       Jacvul  WC3  0.010251687  1.0251687 0.06805547
##       Wave_501   Wave_502
## 184 0.06870667 0.07014220
## 185 0.07235000 0.07368350
## 186 0.06162000 0.06352233
## 187 0.06625000 0.06758350
## 188 0.15275000 0.15415000
## 189 0.06938000 0.07093553
```

```
rm(split_data)

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

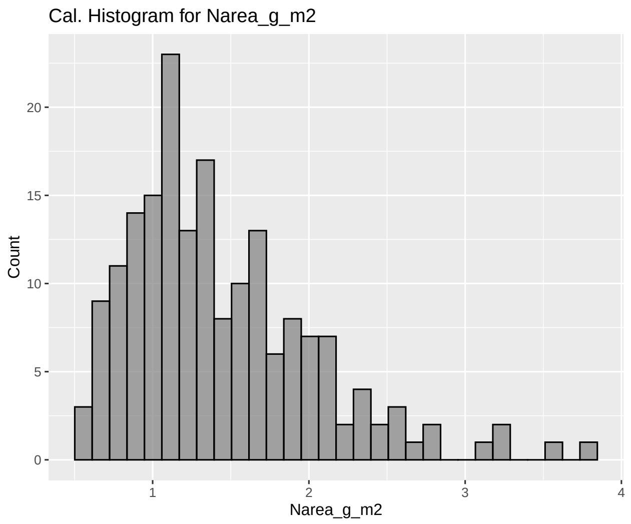
```
## [1] "Cal observations: 183"
```

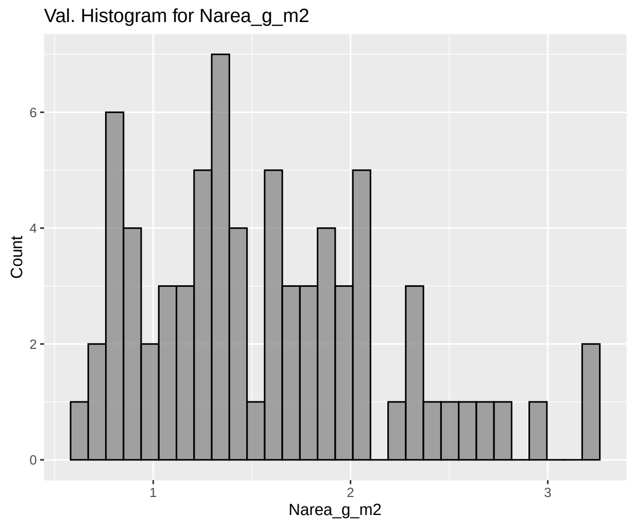
```
print(paste("Val observations: ",dim(val.plsr.data)[1],sep=""))
```

```
## [1] "Val observations: 73"
```

```
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]
```

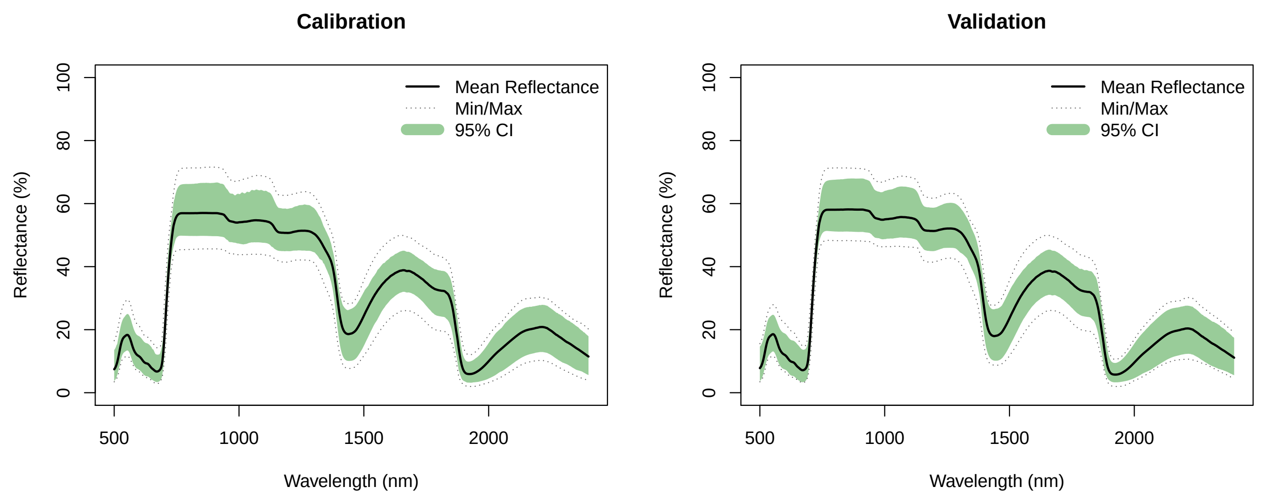
```
##        Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2
## 1 Ammophila arenaria       Ammare  ZC3   0.03240495   3.240495
## 2 Ammophila arenaria       Ammare  MC2   0.02806279   2.806279
## 3 Ammophila arenaria       Ammare  ZC1   0.02041612   2.041612
## 4 Ammophila arenaria       Ammare  MC1   0.02426549   2.426549
## 5 Ammophila arenaria       Ammare  WC3   0.02807281   2.807281
## 6 Ammophila arenaria       Ammare  WR3   0.02286678   2.286678
```

```
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]
```

```
##          Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2
## 184  Jacobaea vulgaris       Jacvul  WC2  0.008756996  0.8756996
## 185 Potentilla reptans       Potrep  WC2  0.010313464  1.0313464
## 186      Rubus caesius       Rubcae  WC2  0.007968454  0.7968454
## 187      Urtica dioica       Urtdio  WC2  0.012737560  1.2737560
## 188 Ammophila arenaria       Ammare  WC3  0.028072806  2.8072806
## 189  Jacobaea vulgaris       Jacvul  WC3  0.010251687  1.0251687
```

plot cal and val spectra

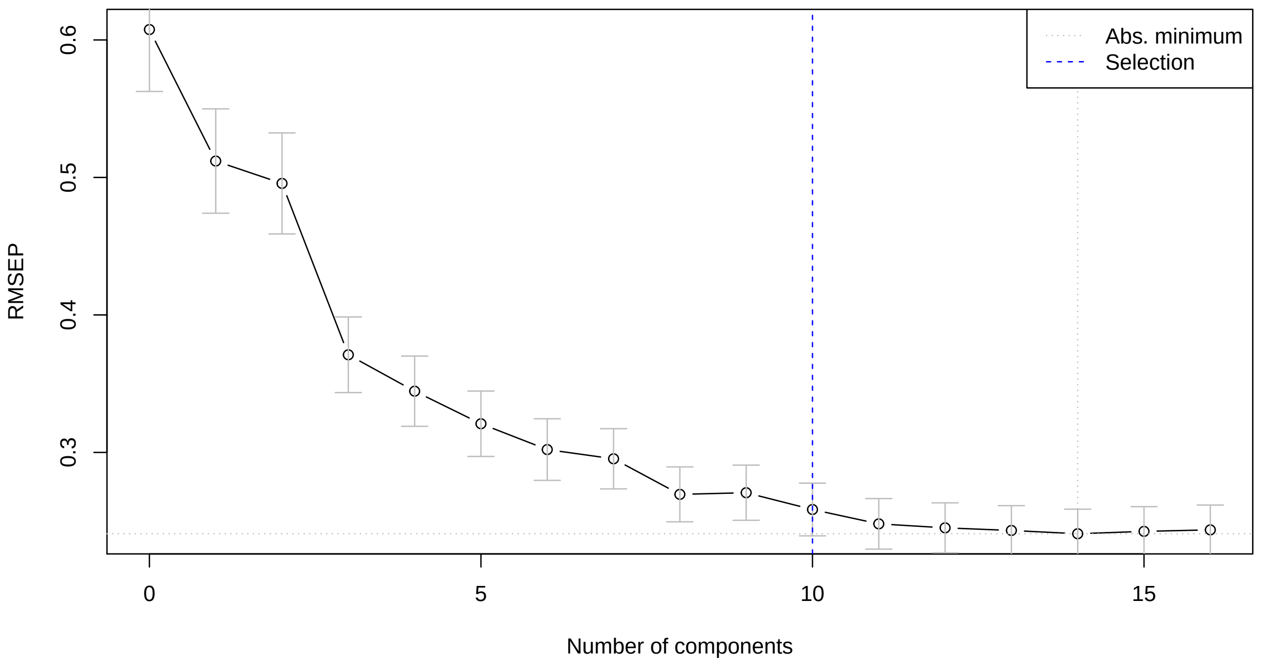
```
par(mfrow=c(1,2)) # B, L, T, R
f.plot.spec(Z=cal.plsr.data$Spectra,wv=seq(Start.wave,End.wave,1),plot_label="Calibration")
f.plot.spec(Z=val.plsr.data$Spectra,wv=seq(Start.wave,End.wave,1),plot_label="Validation")
```



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

Use Jackknife permutation to determine optimal number of components

```r
### 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 <- "pls"
random_seed <- 1245565
seg <- 50
maxComps <- 16
iterations <- 80
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)
} 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)
}
```



Fit final model

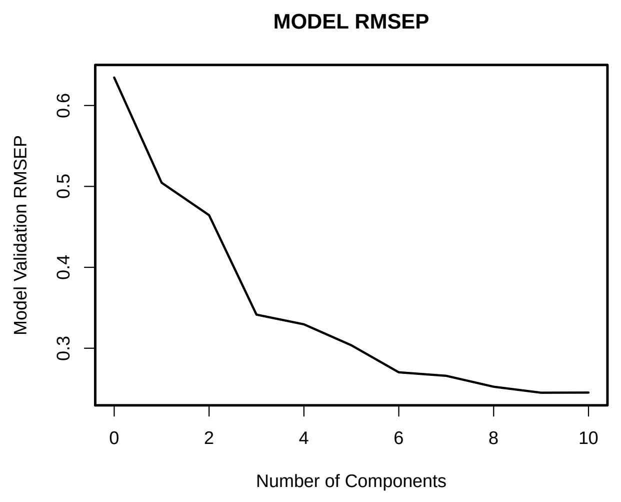
```
### Fit final model - using leave-one-out cross validation
plsr.out <- plsr(as.formula(paste(inVar,"~","Spectra")),scale=FALSE,ncomp=nComps,validation="LOO",
                 trace=FALSE,data=cal.plsr.data)
fit <- plsr.out$fitted.values[,1,nComps]
pls.options(parallel = NULL)

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

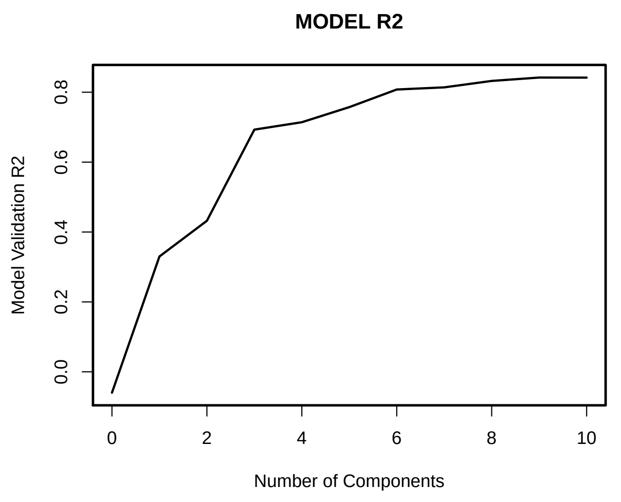
```
## (Intercept)      1 comps      2 comps      3 comps      4 comps      5 comps
##      0.6346       0.5045       0.4645       0.3415       0.3296       0.3037
##     6 comps      7 comps      8 comps      9 comps     10 comps
##      0.2703       0.2659       0.2524       0.2450       0.2452
```

```
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
##    -0.05977      0.33000      0.43217      0.69298      0.71415      0.75732
##     6 comps      7 comps      8 comps      9 comps     10 comps
##     0.80776      0.81389      0.83228      0.84198      0.84176
```

```
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)
```

```
##         Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2 PLSR_Predicted
## 1 Ammophila arenaria       Ammare  ZC3   0.03240495   3.240495       2.672029
## 2 Ammophila arenaria       Ammare  MC2   0.02806279   2.806279       2.651863
## 3 Ammophila arenaria       Ammare  ZC1   0.02041612   2.041612       2.178056
## 4 Ammophila arenaria       Ammare  MC1   0.02426549   2.426549       2.412013
## 5 Ammophila arenaria       Ammare  WC3   0.02807281   2.807281       2.452711
## 6 Ammophila arenaria       Ammare  WR3   0.02286678   2.286678       2.792340
##   PLSR_CV_Predicted PLSR_CV_Residuals
## 1          2.598245      -0.642250440
## 2          2.652066      -0.154212969
## 3          2.200588       0.158975634
## 4          2.435784       0.009234491
## 5          2.384049      -0.423231444
## 6          2.943186       0.656508493
```

```
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)
```

```
##        Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2 PLSR_Predicted
## 1  Jacobaea vulgaris       Jacvul  WC2  0.008756996  0.8756996      0.9462916
## 2 Potentilla reptans       Potrep  WC2  0.010313464  1.0313464      1.5386676
## 3      Rubus caesius       Rubcae  WC2  0.007968454  0.7968454      0.8790482
## 4      Urtica dioica       Urtdio  WC2  0.012737560  1.2737560      1.1241560
## 5 Ammophila arenaria       Ammare  WC3  0.028072806  2.8072806      2.4527108
## 6  Jacobaea vulgaris       Jacvul  WC3  0.010251687  1.0251687      1.1553688
##   PLSR_Residuals
## 1     0.07059201
## 2     0.50732119
## 3     0.08220284
## 4    -0.14959995
## 5    -0.35456980
## 6     0.13020008
```

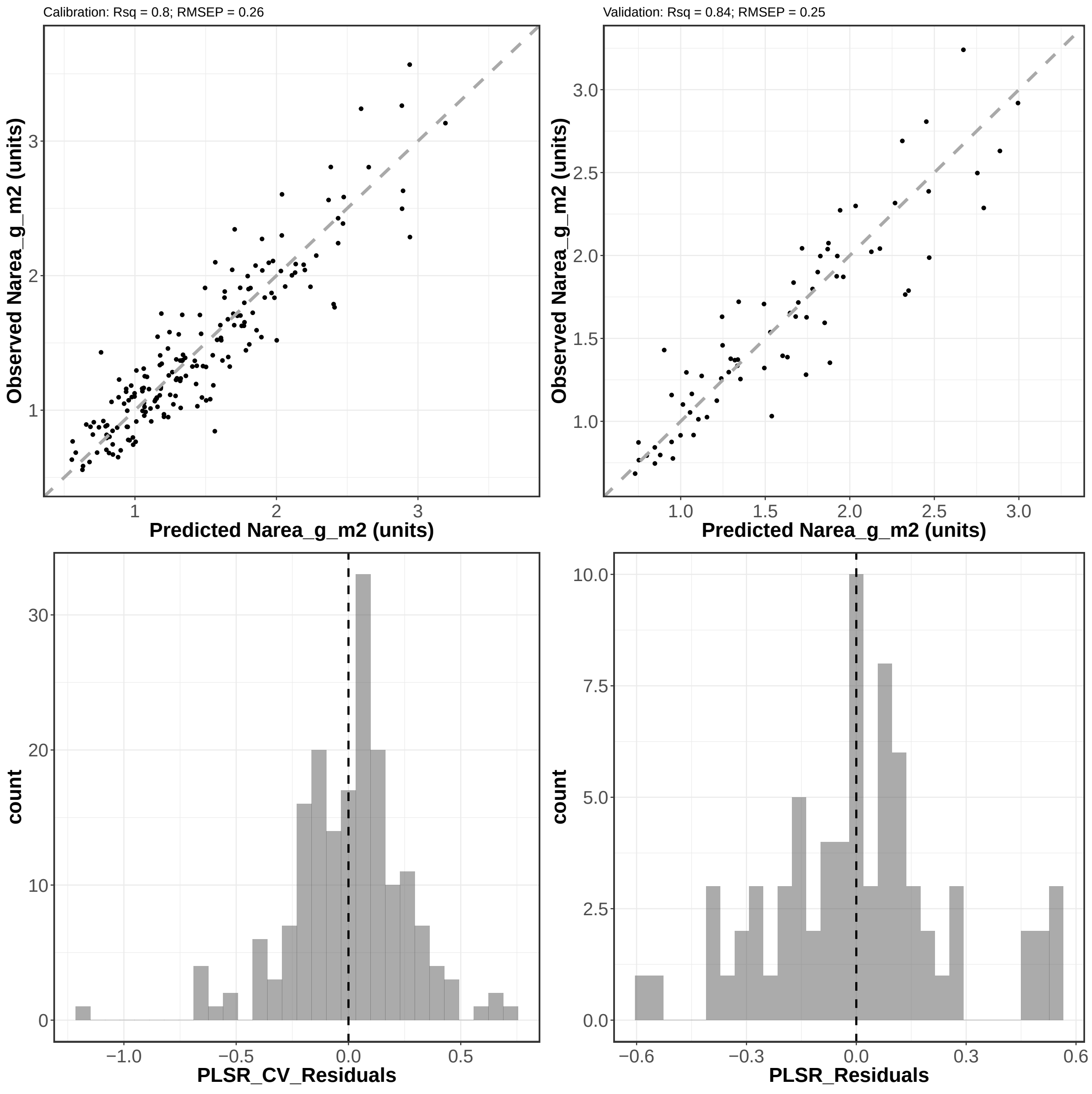
```
val.R2 <- round(pls::R2(plsr.out,newdata=val.plsr.data)[[1]][nComps],2)
val.RMSEP <- round(sqrt(mean(val.plsr.output$PLSR_Residuals^2)),2)

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))

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("Rsq = ", val.R2), "; ", paste0("RMSEP = ",
                                                                          val.RMSEP))) +
  theme(axis.text=element_text(size=18), legend.position="none",
        axis.title=element_text(size=20, face="bold"),
```

```r
        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 2 rows containing missing values (geom_point).
```

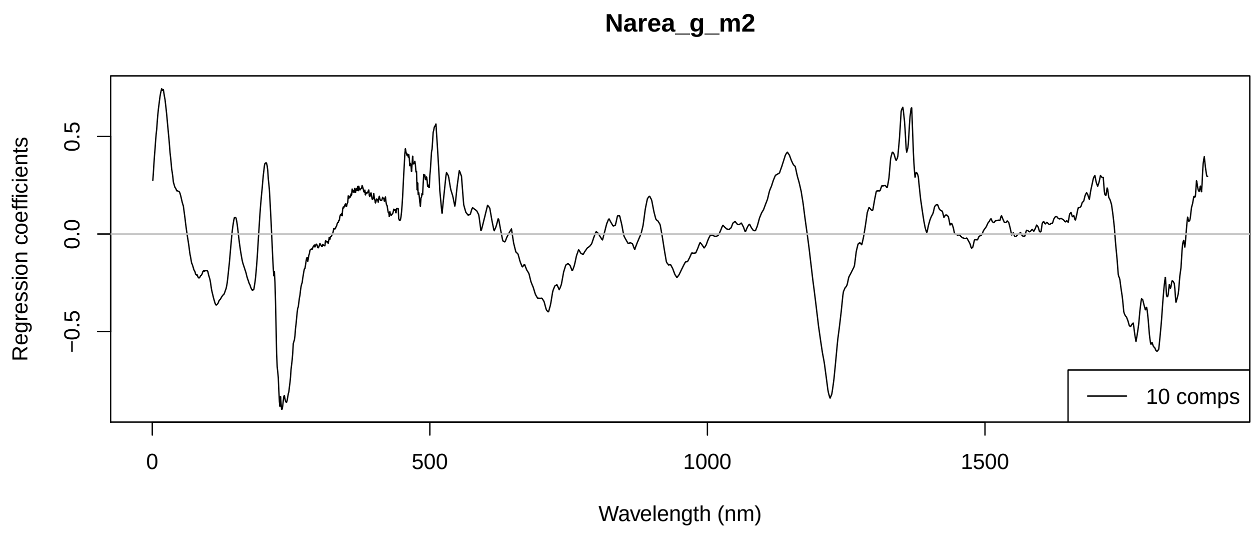
```
## Warning: Removed 3 rows containing missing values (geom_point).
```

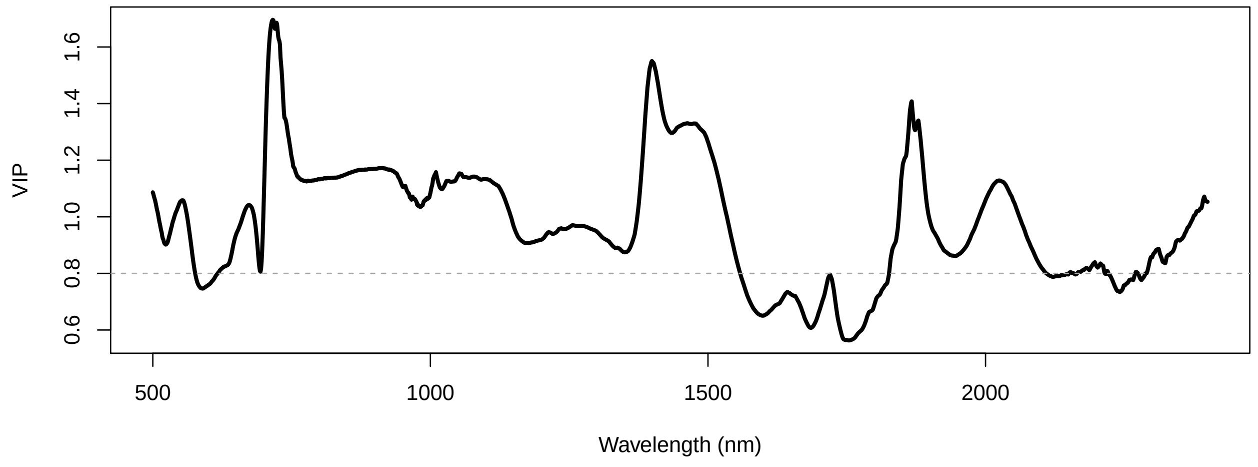
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



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

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

jk.plsr.out <- pls::plsr(as.formula(paste(inVar,"~","Spectra")), scale=FALSE, center=TRUE, ncomp=nComps
                         validation="LOO", trace=TRUE, jackknife=TRUE, data=cal.plsr.data)
```

```
## Segment:
```

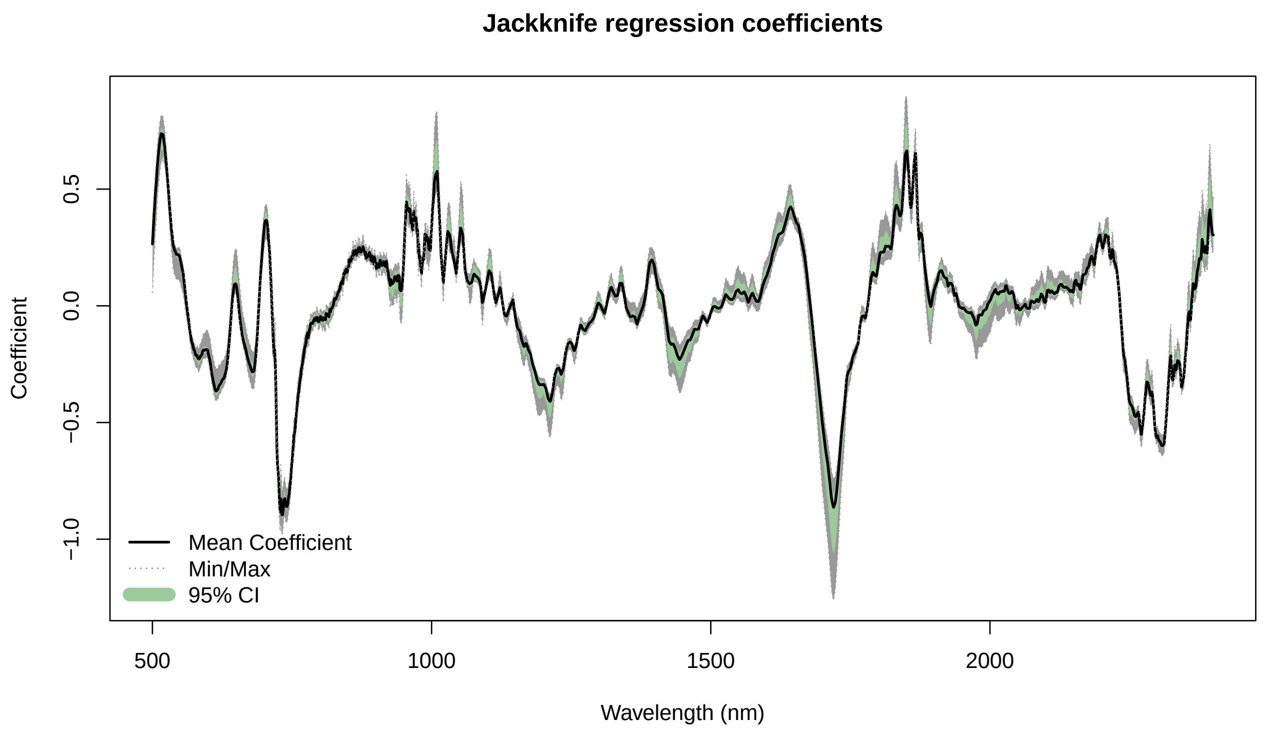
```r
pls.options(parallel = NULL)

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],,,]
```

```r
#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)
```

```
##        Plant_Species Species_Code Plot Narea_mg_mm2 Narea_g_m2 PLSR_Predicted
## 1  Jacobaea vulgaris       Jacvul  WC2  0.008756996  0.8756996      0.9462916
## 2 Potentilla reptans       Potrep  WC2  0.010313464  1.0313464      1.5386676
## 3      Rubus caesius       Rubcae  WC2  0.007968454  0.7968454      0.8790482
## 4      Urtica dioica       Urtdio  WC2  0.012737560  1.2737560      1.1241560
## 5 Ammophila arenaria       Ammare  WC3  0.028072806  2.8072806      2.4527108
## 6  Jacobaea vulgaris       Jacvul  WC3  0.010251687  1.0251687      1.1553688
##   PLSR_Residuals       LCI      UCI       LPI      UPI
## 1     0.07059201 0.7931248 1.079405 0.4390577 1.453526
## 2     0.50732119 1.3677140 1.670654 1.0274118 2.049923
## 3     0.08220284 0.7692936 1.013202 0.3770260 1.381070
## 4    -0.14959995 1.0191721 1.258393 0.6226488 1.625663
## 5    -0.35456980 2.3627510 2.590288 1.9515079 2.953914
## 6     0.13020008 1.0085494 1.285158 0.6483477 1.662390
```

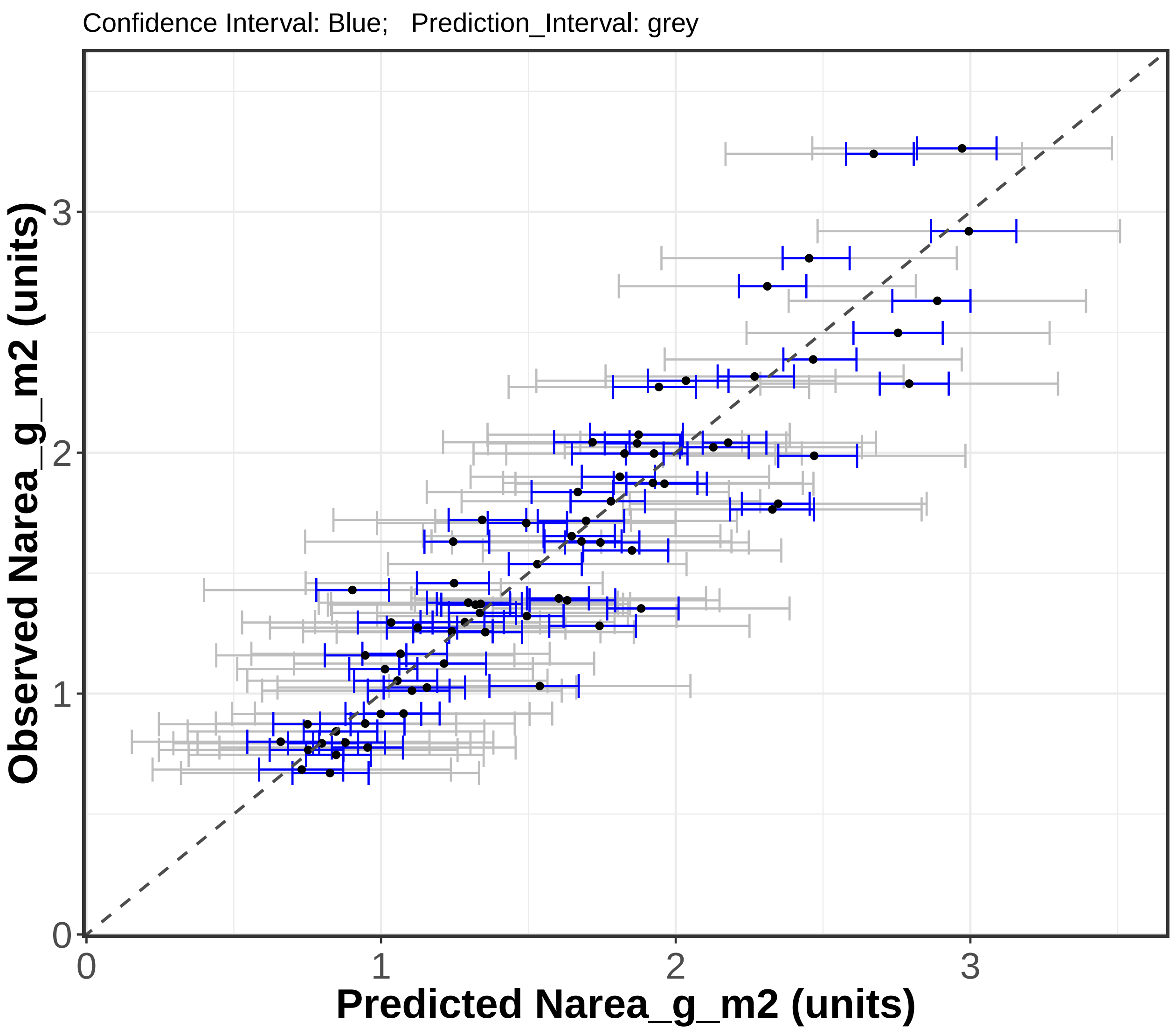
Jackknife coefficient plot

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



Jackknife validation plot

```
#rng_vals <- quantile(val.plsr.output[,inVar], probs = c(0.001, 0.999))
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=0.1) +
  geom_errorbar(aes(xmin = LCI,xmax = UCI),color='blue',width=0.1) +
  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))
jk_val_scatterplot
```



Output jackknife results

```
out.jk.coefs <- data.frame(Iteration=seq(1,length(Jackknife_intercept),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 -0.001089661 0.3156927 0.3524556 0.3947195 0.4329382
## Seg 2          2  0.082969588 0.2989509 0.3382983 0.3835509 0.4239103
## Seg 3          3  0.114879574 0.2716867 0.3122469 0.3574386 0.3982935
## Seg 4          4  0.178884696 0.2099486 0.2520760 0.3018899 0.3452178
## Seg 5          5  0.126339690 0.2898707 0.3311239 0.3762377 0.4163999
## Seg 6          6 -0.085381533 0.2805890 0.3195387 0.3625074 0.4023830
```

```
write.csv(out.jk.coefs,file=file.path(outdir,
                                      paste0(inVar,
                                             '_Jackkife_PLSR_Coefficients.csv')),
          row.names=FALSE)
```

Create core PLSR outputs

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

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

```r
# 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

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

```
## [1] "**** PLSR output files: "
```

```r
list.files(outdir)[grep(pattern = inVar, list.files(outdir))]
```

```
## [1] "Narea_g_m2_Jackkife_PLSR_Coefficients.csv"
## [2] "Narea_g_m2_Observed_PLSR_CV_Pred_10comp.csv"
## [3] "Narea_g_m2_PLSR_Coefficients_10comp.csv"
## [4] "Narea_g_m2_PLSR_VIPs_10comp.csv"
## [5] "Narea_g_m2_Validation_PLSR_Pred_10comp.csv"
```