

# Spectra-trait PLSR example using leaf-level spectra and leaf nitrogen content (Narea, g/m<sup>2</sup>) data from eight different crop species growing in a glasshouse at Brookhaven National Laboratory. This example illustrates running the PLSR permutation by group

Shawn P. Serbin, Julien Lamour, & Jeremiah Anderson

2025-10-29

## Overview

This is an R Markdown Notebook to illustrate how to load an internal dataset (“ely\_plsr\_data”), choose the “optimal” number of pls components, and fit a pls model for leaf nitrogen content (Narea, g/m<sup>2</sup>)

## Getting Started

### Load libraries

```
list.of.packages <- c("pls", "dplyr", "here", "plotrix", "ggplot2", "gridExtra", "spectratrait")
invisible(lapply(list.of.packages, library, character.only = TRUE))

##
## 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/spectratrait
##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:dplyr':
##
##   combine
```

## Setup other functions and options

```
### Setup options

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

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

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

# Specify output directory, output_dir
# Options:
# tempdir - use a OS-specified temporary directory
# user defined PATH - e.g. "~/scratch/PLSR"
output_dir <- "tempdir"
```

## Load internal Ely et al 2019 dataset

```
data("ely_plsr_data")
head(ely_plsr_data)[,1:8]
```

```
##   Species_Code      Common_Name C_N_mass  C_g_m2 H2O_g_m2 LMA_g_m2  N_g_m2
## 1          HEAN3 common sunflower    7.58 15.61210  167.63   36.40 2.103694
## 2          HEAN3 common sunflower    8.33 14.73724  164.68   34.65 1.231713
## 3          HEAN3 common sunflower    7.70 15.02495  156.95   35.08 1.764752
## 4          CUSA4 garden cucumber    7.40 11.14835  111.52   26.23 1.287963
## 5          CUSA4 garden cucumber    7.47 11.60735  123.58   26.71 1.411361
## 6          CUSA4 garden cucumber    7.43  8.06035  114.36   18.40 1.117704
##   Wave_500
## 1 4.782000
## 2 4.341714
## 3 4.502857
## 4 3.333429
## 5 3.313571
## 6 3.272286
```

```
# What is the target variable?
inVar <- "N_g_m2"
```

## Set working directory (scratch space)

```
## [1] "/private/var/folders/tq/tydmhlwn1bdf_0pmpcq70r2c0000gn/T/RtmpAcMfZ9"
```

## Full PLSR dataset

```
Start.wave <- 500
End.wave <- 2400
wv <- seq(Start.wave, End.wave, 1)
pls_data <- ely_plsr_data
head(pls_data)[,1:6]
```

	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2	LMA_g_m2
## 1	HEAN3	common sunflower	7.58	15.61210	167.63	36.40
## 2	HEAN3	common sunflower	8.33	14.73724	164.68	34.65
## 3	HEAN3	common sunflower	7.70	15.02495	156.95	35.08
## 4	CUSA4	garden cucumber	7.40	11.14835	111.52	26.23
## 5	CUSA4	garden cucumber	7.47	11.60735	123.58	26.71
## 6	CUSA4	garden cucumber	7.43	8.06035	114.36	18.40

Create cal/val datasets

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

method <- "base" #base/dplyr
# base R - a bit slow
# dplyr - much faster
split_data <- spectratrait::create_data_split(dataset=plsr_data, approach=method,
                                              split_seed=23452135, prop=0.7,
                                              group_variables="Species_Code")
```

```
## HEAN3   Cal: 70%
## CUSA4   Cal: 68.182%
## CUPE    Cal: 70.588%
## SOLYL   Cal: 70%
## OCBA    Cal: 68.421%
## POPUL   Cal: 71.429%
## GLMA4   Cal: 70.588%
## PHVU    Cal: 66.667%
```

```
names(split_data)
```

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

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

	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2	LMA_g_m2	N_g_m2
## 1	HEAN3	common sunflower	7.58	15.61210	167.63	36.40	2.103694
## 2	HEAN3	common sunflower	8.33	14.73724	164.68	34.65	1.231713
## 4	CUSA4	garden cucumber	7.40	11.14835	111.52	26.23	1.287963
## 6	CUSA4	garden cucumber	7.43	8.06035	114.36	18.40	1.117704
## 7	CUPE	field pumpkin	7.20	11.43007	128.42	25.83	1.215333
## 10	SOLYL	garden tomato	7.89	11.61918	142.23	27.40	1.304110
##	Wave_500						
## 1	4.782000						
## 2	4.341714						
## 4	3.333429						
## 6	3.272286						
## 7	2.943143						
## 10	4.145714						

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

```
##   Species_Code      Common_Name C_N_mass   C_g_m2 H2O_g_m2 LMA_g_m2   N_g_m2
## 3      HEAN3 common sunflower    7.70 15.024947  156.95   35.08 1.7647515
## 5      CUSA4  garden cucumber    7.47 11.607347  123.58   26.71 1.4113615
## 8      CUPE   field pumpkin     7.67 12.466238  124.67   29.22 1.1468413
## 9      CUPE   field pumpkin     7.64 17.100448  142.85   43.39 1.1390174
## 13     SOLYL  garden tomato     7.73  7.938866  129.95   17.96 0.9483533
## 15     OCBA   sweet basil       8.13 16.975969  173.30   38.65 1.1246459
##   Wave_500
## 3  4.502857
## 5  3.313571
## 8  2.868000
## 9  3.338286
## 13 3.960286
## 15 3.744000
```

```
rm(split_data)
```

```
# Datasets:
```

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

```
## [1] "Cal observations: 124"
```

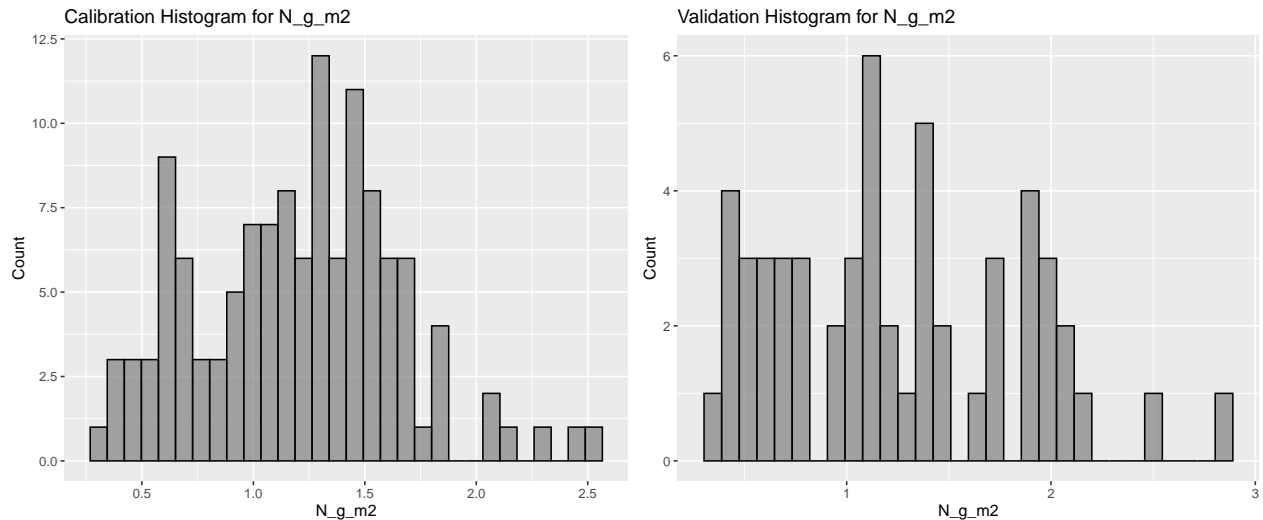
```
print(paste("Val observations: ",dim(val.plsr.data)[1],sep=""))
```

```
## [1] "Val observations: 54"
```

```
cal_hist_plot <- ggplot(data = cal.plsr.data,
                        aes(x = cal.plsr.data[,paste0(inVar)])) +
  geom_histogram(fill=I("grey50"),col=I("black"),alpha=I(.7)) +
  labs(title=paste0("Calibration Histogram for ",inVar), x = paste0(inVar),
       y = "Count")
val_hist_plot <- ggplot(data = val.plsr.data,
                        aes(x = val.plsr.data[,paste0(inVar)])) +
  geom_histogram(fill=I("grey50"),col=I("black"),alpha=I(.7)) +
  labs(title=paste0("Validation Histogram for ",inVar), x = paste0(inVar),
       y = "Count")
histograms <- 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`.
```



```
ggsave(filename = file.path(outdir,paste0(inVar,"_Cal_Val_Histograms.png")),
  plot = histograms,
  device="png", width = 30,
  height = 12, units = "cm",
  dpi = 300)
# output cal/val data
write.csv(cal.plsr.data,file=file.path(outdir,paste0(inVar,'_Cal_PLSR_Dataset.csv')),
  row.names=FALSE)
write.csv(val.plsr.data,file=file.path(outdir,paste0(inVar,'_Val_PLSR_Dataset.csv')),
  row.names=FALSE)
```

## 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]
```

##	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2
## 1	HEAN3	common sunflower	7.58	15.61210	167.63
## 2	HEAN3	common sunflower	8.33	14.73724	164.68
## 4	CUSA4	garden cucumber	7.40	11.14835	111.52
## 6	CUSA4	garden cucumber	7.43	8.06035	114.36
## 7	CUPE	field pumpkin	7.20	11.43007	128.42
## 10	SOLYL	garden tomato	7.89	11.61918	142.23

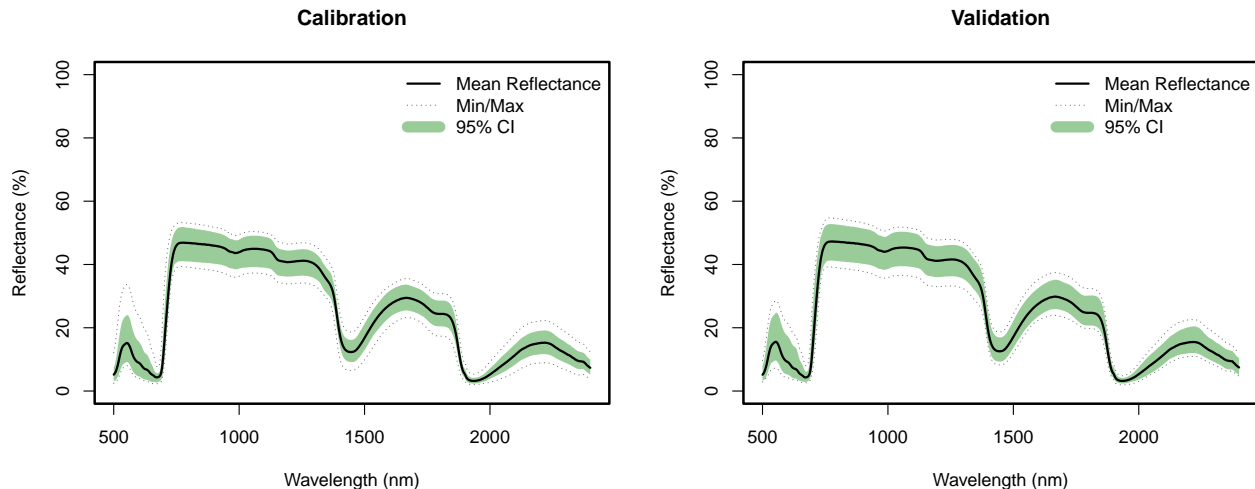
```
val_spec <- as.matrix(val.plsr.data[, which(names(val.plsr.data) %in% paste0("Wave_",wv))])
val.plsr.data <- data.frame(val.plsr.data[, which(names(val.plsr.data) %notin% paste0("Wave_",wv))],
  Spectra=I(val_spec))
head(val.plsr.data)[1:5]
```

##	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2
## 3	HEAN3	common sunflower	7.70	15.024947	156.95
## 5	CUSA4	garden cucumber	7.47	11.607347	123.58
## 8	CUPE	field pumpkin	7.67	12.466238	124.67
## 9	CUPE	field pumpkin	7.64	17.100448	142.85
## 13	SOLYL	garden tomato	7.73	7.938866	129.95

```
## 15          OCBA          sweet basil          8.13 16.975969 173.30
```

plot cal and val spectra

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



```
dev.copy(png,file.path(outdir,paste0(inVar,'_Cal_Val_Spectra.png')),
         height=2500,width=4900, res=340)
```

```
## quartz_off_screen
##          3
```

```
dev.off();
```

```
## pdf
##    2
```

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

Use 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 <- "firstMin" #firstPlateau, firstMin
random_seed <- 1245565
seg <- 50
maxComps <- 16
iterations <- 80
prop <- 0.70
nComps <- spectratrait::find_optimal_comp_by_groups(dataset=cal.plsr.data, targetVariable=inVar,
                                                    method=method, maxComps=maxComps,
                                                    iterations=iterations, prop=prop,
```

```

random_seed=random_seed,
group_variables="Species_Code")

```

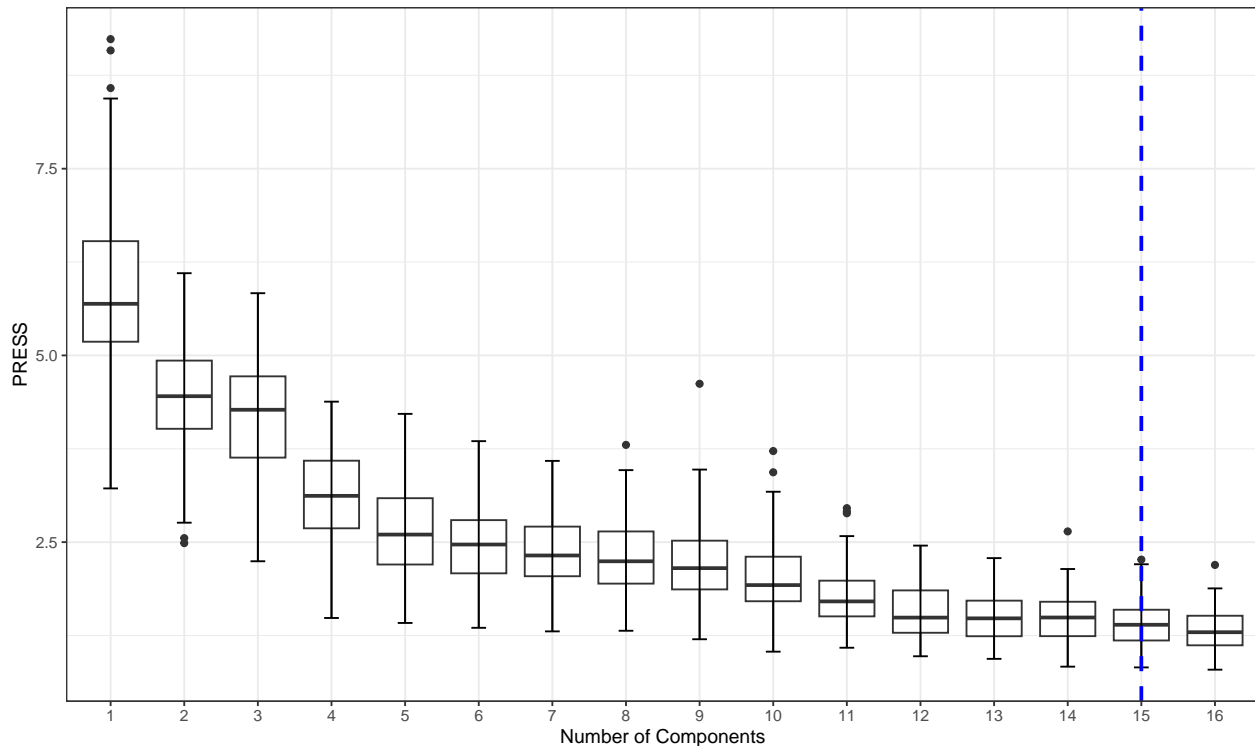
```

## [1] "*** Identifying optimal number of PLSR components using stratified resampling by group_variables"
## [1] "*** Running permutation test. Please hang tight, this can take awhile ***"
## [1] "Options:"
## [1] "Max Components: 16 Iterations: 80 Data Proportion (percent): 70"
## [1] "*** Providing PRESS and coefficient array output ***"

## No id variables; using all as measure variables

## [1] "*** Optimal number of components based on t.test: 15"

```



```

dev.copy(png,file.path(outdir,paste0(paste0(inVar,"_PLSR_Component_Selection.png"))),
height=2800, width=3400, res=340)

```

```

## quartz_off_screen
## 3

```

```

dev.off();

```

```

## pdf
## 2

```

## Fit final model

```

plsr.out <- plsr(as.formula(paste(inVar,"~","Spectra")),scale=FALSE,ncomp=nComps,validation="L00",
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
pls::RMSEP(plsr.out, newdata = val.plsr.data)

## (Intercept)      1 comps      2 comps      3 comps      4 comps      5 comps
##      0.5908      0.4735      0.4162      0.4037      0.3347      0.3023
##      6 comps      7 comps      8 comps      9 comps     10 comps     11 comps
##      0.2993      0.3081      0.2814      0.2445      0.2276      0.2104
##     12 comps     13 comps     14 comps     15 comps
##      0.1954      0.2003      0.1973      0.2108

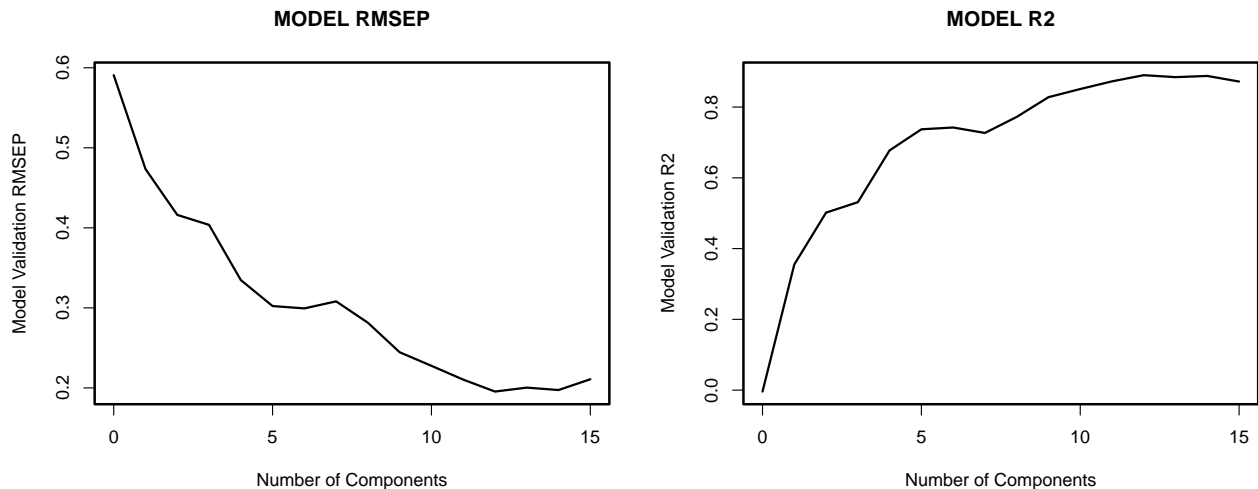
plot(pls::RMSEP(plsr.out,estimate=c("test"),newdata = val.plsr.data), main="MODEL RMSEP",
     xlab="Number of Components",ylab="Model Validation RMSEP",lty=1,col="black",cex=1.5,lwd=2)
box(lwd=2.2)

pls::R2(plsr.out, newdata = val.plsr.data)

## (Intercept)      1 comps      2 comps      3 comps      4 comps      5 comps
##     -0.004079     0.355010     0.501632     0.531088     0.677620     0.737143
##      6 comps      7 comps      8 comps      9 comps     10 comps     11 comps
##      0.742224     0.726835     0.772115     0.827942     0.850962     0.872685
##     12 comps     13 comps     14 comps     15 comps
##      0.890124     0.884529     0.887961     0.872129

plot(pls::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)

```



```

dev.copy(png,file.path(outdir,paste0(paste0(inVar,"_Validation_RMSEP_R2_by_Component.png"))),
         height=2800, width=4800, res=340)

## quartz_off_screen
##      3

dev.off();

## pdf
##      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)
```

##	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2	LMA_g_m2	N_g_m2
## 1	HEAN3	common sunflower	7.58	15.61210	167.63	36.40	2.103694
## 2	HEAN3	common sunflower	8.33	14.73724	164.68	34.65	1.231713
## 4	CUSA4	garden cucumber	7.40	11.14835	111.52	26.23	1.287963
## 6	CUSA4	garden cucumber	7.43	8.06035	114.36	18.40	1.117704
## 7	CUPE	field pumpkin	7.20	11.43007	128.42	25.83	1.215333
## 10	SOLYL	garden tomato	7.89	11.61918	142.23	27.40	1.304110

```
##      PLSR_Predicted PLSR_CV_Predicted PLSR_CV_Residuals
## 1      1.836047      1.714086      -0.38960842
## 2      1.530813      1.685388       0.45367526
## 4      1.254794      1.262835      -0.02512724
## 6      1.127053      1.129340       0.01163542
## 7      1.196259      1.188471      -0.02686200
## 10     1.276380      1.281683      -0.02242624

cal.R2 <- round(pls::R2(plsr.out,intercept=F)[[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)
```

##	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2	LMA_g_m2	N_g_m2
## 3	HEAN3	common sunflower	7.70	15.024947	156.95	35.08	1.7647515
## 5	CUSA4	garden cucumber	7.47	11.607347	123.58	26.71	1.4113615
## 8	CUPE	field pumpkin	7.67	12.466238	124.67	29.22	1.1468413
## 9	CUPE	field pumpkin	7.64	17.100448	142.85	43.39	1.1390174
## 13	SOLYL	garden tomato	7.73	7.938866	129.95	17.96	0.9483533
## 15	OCBA	sweet basil	8.13	16.975969	173.30	38.65	1.1246459

```
##      PLSR_Predicted PLSR_Residuals
## 3      1.7624701   -0.002281391
## 5      1.2947218   -0.116639722
## 8      0.9934199   -0.153421396
## 9      1.1345273   -0.004490078
## 13     0.7432855   -0.205067758
## 15     1.1613789    0.036733007

val.R2 <- round(pls::R2(plsr.out,newdata=val.plsr.data,intercept=F)[[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", linewidth=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("Rsqr = ", 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, linewidth=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", linewidth=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, linewidth=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", linewidth=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, linewidth=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", linewidth=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, linewidth=1.5))

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

```

```

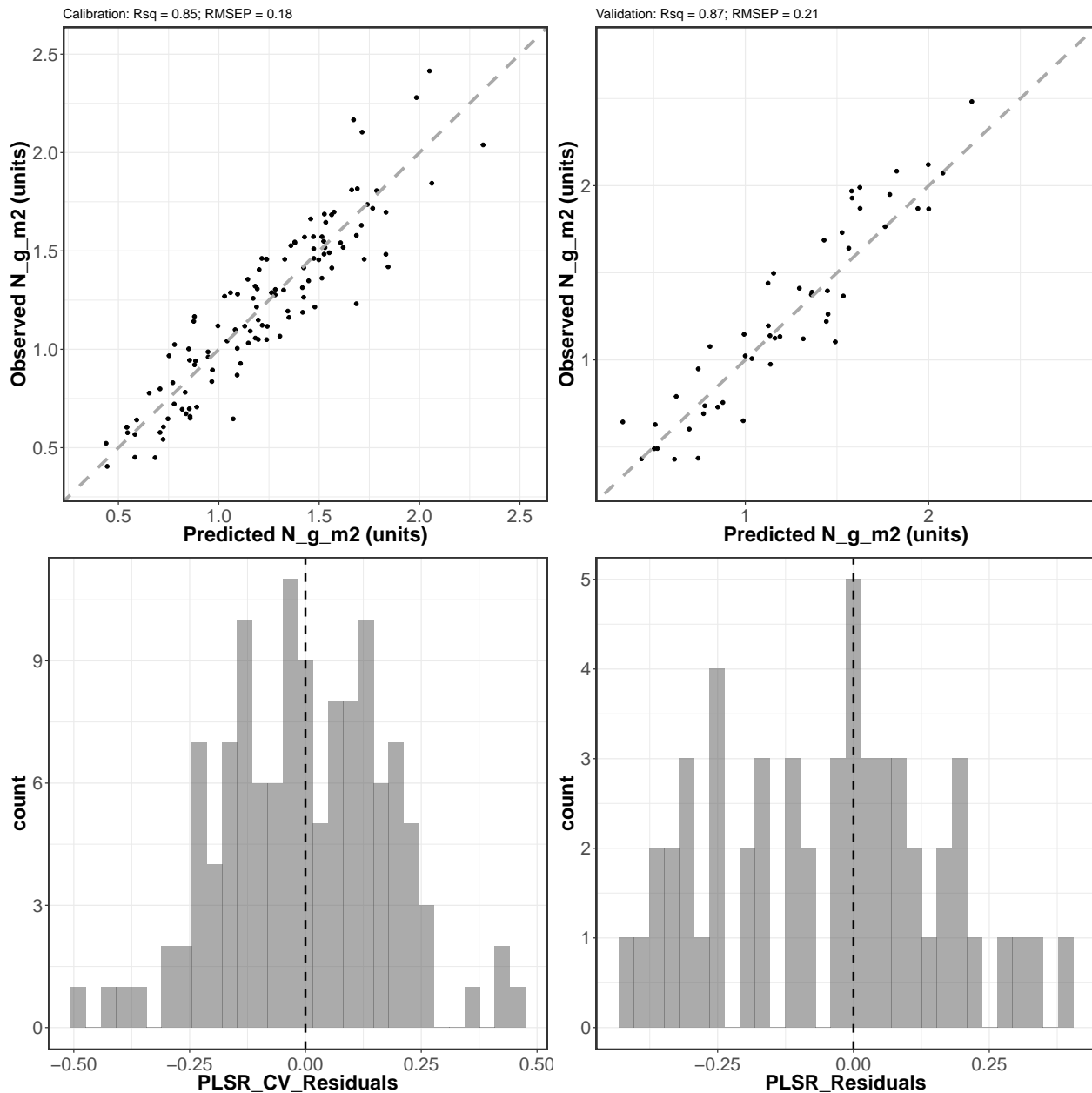
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).

```

```
## Warning: Removed 4 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

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

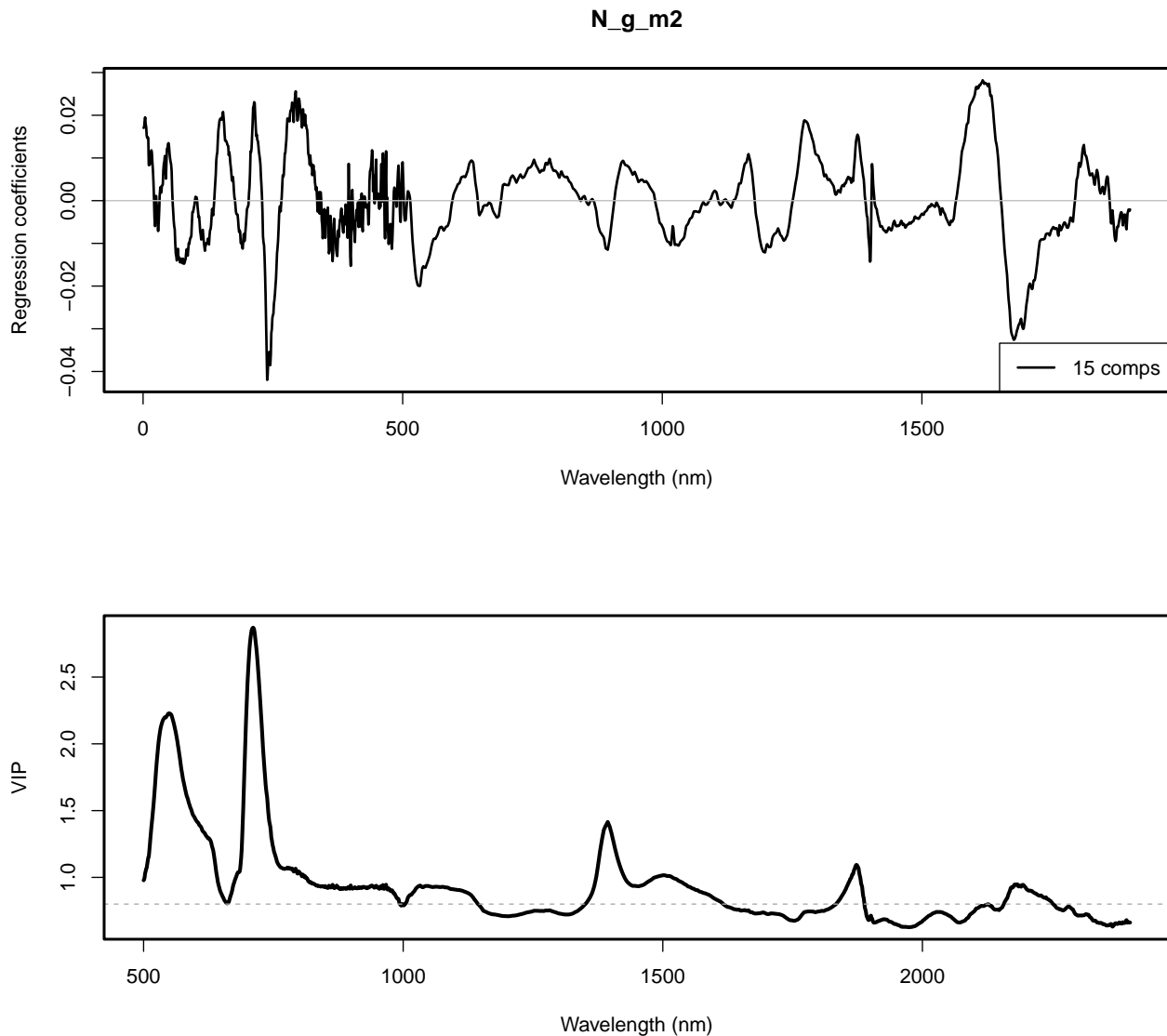
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
ggsave(filename = file.path(outdir,paste0(inVar,"_Cal_Val_Scatterplots.png")),
        plot = scatterplots, device="png",
        width = 32,
        height = 30, units = "cm",
        dpi = 300)
```

## Generate Coefficient and VIP plots

```
vips <- spectratrait::VIP(plsr.out)[nComps,]
par(mfrow=c(2,1))
plot(plsr.out, plotype = "coef",xlab="Wavelength (nm)",
     ylab="Regression coefficients",legendpos = "bottomright",
     ncomp=nComps,lwd=2)
box(lwd=2.2)
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")
box(lwd=2.2)
```



```
dev.copy(png,file.path(outdir,paste0(inVar,'_Coefficient_VIP_plot.png')),
        height=3100, width=4100, res=340)
```

```
## quartz_off_screen
## 3
```

```
dev.off();
```

```
## pdf
## 2
```

## Bootstrap validation

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

### PLSR bootstrap permutation uncertainty analysis
iterations <- 500 # how many permutation iterations to run
prop <- 0.70 # fraction of training data to keep for each iteration
plsr_permutation <- spectratrait::pls_permutation_by_groups(dataset=cal.plsr.data,
  targetVariable=inVar,
  maxComps=nComps,
  iterations=iterations,
  prop=prop, group_variables="Species_Code",
  verbose=FALSE)

## [1] "*** Running permutation test. Please hang tight, this can take awhile ***"
## [1] "Options:"
## [1] "Max Components: 15 Iterations: 500 Data Proportion (percent): 70"
## [1] "*** Providing PRESS and coefficient array output ***"

bootstrap_intercept <- pls_permutation$coef_array[1,nComps]
bootstrap_coef <- pls_permutation$coef_array[2:length(pls_permutation$coef_array[,1,nComps]),
  ,nComps]

rm(pls_permutation)

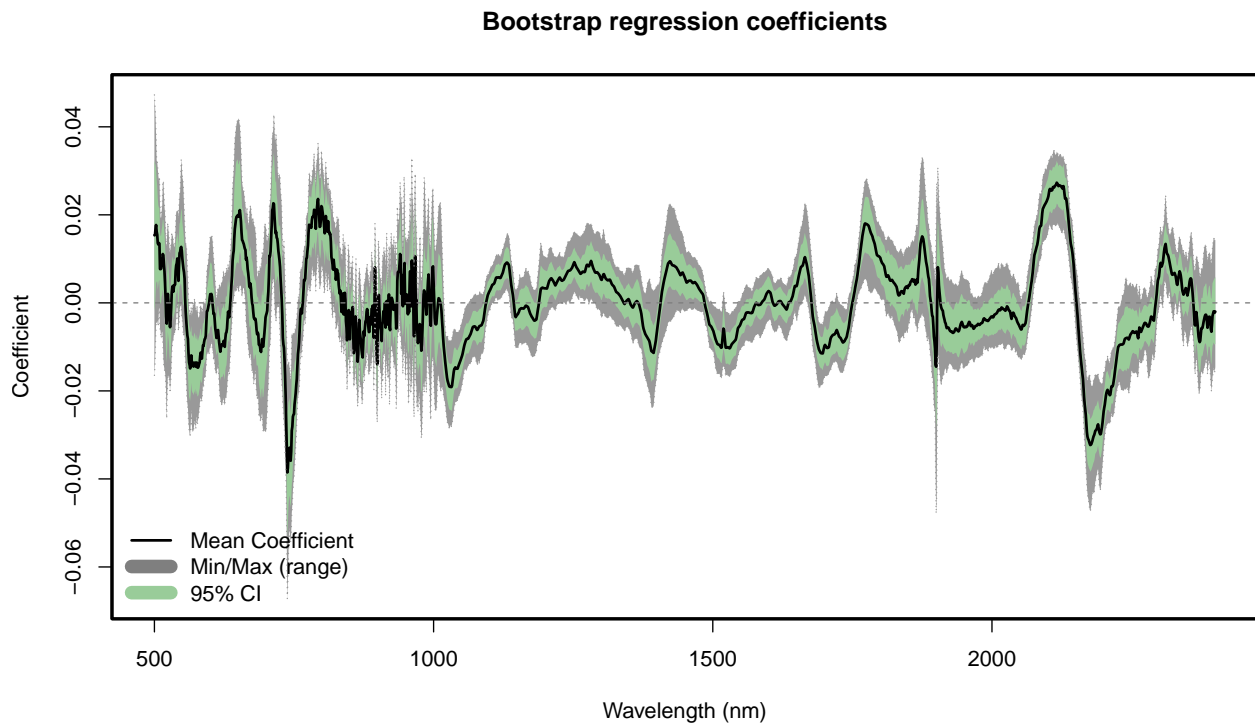
# apply coefficients to left-out validation data
interval <- c(0.025,0.975)
Bootstrap_Pred <- val.plsr.data$Spectra %*% bootstrap_coef +
  matrix(rep(bootstrap_intercept, length(val.plsr.data[,inVar])), byrow=TRUE,
    ncol=length(bootstrap_intercept))
Interval_Conf <- apply(X = Bootstrap_Pred, MARGIN = 1, FUN = quantile,
  probs=c(interval[1], interval[2]))
sd_mean <- apply(X = Bootstrap_Pred, MARGIN = 1, FUN = sd)
sd_res <- sd(val.plsr.output$PLSR_Residuals)
sd_tot <- sqrt(sd_mean^2+sd_res^2)
val.plsr.output$LCI <- Interval_Conf[1,]
val.plsr.output$UCI <- Interval_Conf[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)
```

##	Species_Code	Common_Name	C_N_mass	C_g_m2	H2O_g_m2	LMA_g_m2	N_g_m2
## 3	HEAN3	common sunflower	7.70	15.024947	156.95	35.08	1.7647515
## 5	CUSA4	garden cucumber	7.47	11.607347	123.58	26.71	1.4113615
## 8	CUPE	field pumpkin	7.67	12.466238	124.67	29.22	1.1468413
## 9	CUPE	field pumpkin	7.64	17.100448	142.85	43.39	1.1390174
## 13	SOLYL	garden tomato	7.73	7.938866	129.95	17.96	0.9483533

```
## 15      OCBA      sweet basil      8.13 16.975969   173.30   38.65 1.1246459
##      PLSR_Predicted PLSR_Residuals      LCI      UCI      LPI      UPI
## 3      1.7624701   -0.002281391 1.5710330 1.9443661 1.3151243 2.209816
## 5      1.2947218   -0.116639722 1.2019841 1.4531979 0.8688563 1.720587
## 8      0.9934199   -0.153421396 0.8544582 1.1646561 0.5564158 1.430424
## 9      1.1345273   -0.004490078 0.9954061 1.2824287 0.7007745 1.568280
## 13     0.7432855   -0.205067758 0.5836738 0.9094675 0.3042086 1.182362
## 15     1.1613789    0.036733007 1.0021191 1.2849671 0.7291004 1.593657
```

### Jackknife coefficient plot

```
# Bootstrap regression coefficient plot
spectratrait::f.plot.coef(Z = t(bootstrap_coef), wv = wv,
  plot_label="Bootstrap regression coefficients", position = 'bottomleft')
abline(h=0,lty=2,col="grey50")
box(lwd=2.2)
```



```
dev.copy(png,file.path(outdir,paste0(inVar, '_Bootstrap_Regression_Coefficients.png')),
  height=2100, width=3800, res=340)
```

```
## quartz_off_screen
##      3
```

```
dev.off();
```

```
## pdf
##      2
```

### Bootstrap validation plot

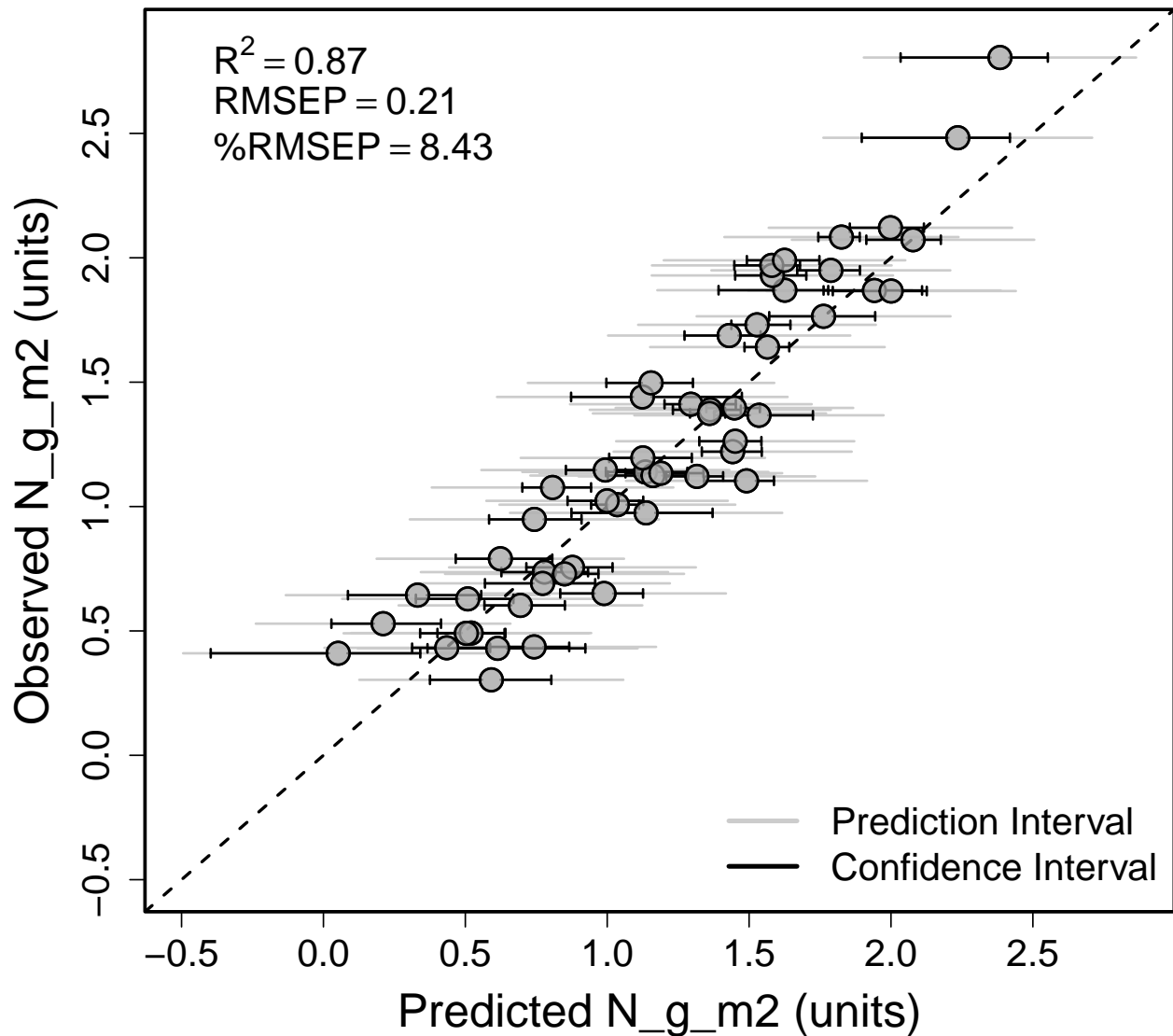
```
rmsep_percrmsep <- spectratrait::percent_rmse(plsr_dataset = val.plsr.output,
  inVar = inVar,
```

```

residuals = val.plsr.output$PLSR_Residuals,
range="full")

RMSEP <- rmsep_percrmsep$rmse
perc_RMSEP <- rmsep_percrmsep$perc_rmse
r2 <- round(pls::R2(plsr.out, newdata = val.plsr.data, intercept=F)$val[nComps],2)
expr <- vector("expression", 3)
expr[[1]] <- bquote(R^2==.(r2))
expr[[2]] <- bquote(RMSEP==.(round(RMSEP,2)))
expr[[3]] <- bquote("%RMSEP"==.(round(perc_RMSEP,2)))
rng_vals <- c(min(val.plsr.output$LPI), max(val.plsr.output$UPI))
par(mfrow=c(1,1), mar=c(4.2,5.3,1,0.4), oma=c(0, 0.1, 0, 0.2))
plotrix::plotCI(val.plsr.output$PLSR_Predicted, val.plsr.output[,inVar],
  li=val.plsr.output$LPI, ui=val.plsr.output$UPI, gap=0.009, sfrac=0.000,
  lwd=1.6, xlim=c(rng_vals[1], rng_vals[2]), ylim=c(rng_vals[1], rng_vals[2]),
  err="x", pch=21, col="black", pt.bg=scales::alpha("grey70",0.7), scol="grey80",
  cex=2, xlab=paste0("Predicted ", paste(inVar), " (units)"),
  ylab=paste0("Observed ", paste(inVar), " (units)"),
  cex.axis=1.5, cex.lab=1.8)
abline(0,1,lty=2,lw=2)
plotrix::plotCI(val.plsr.output$PLSR_Predicted, val.plsr.output[,inVar],
  li=val.plsr.output$LCI, ui=val.plsr.output$UCI, gap=0.009, sfrac=0.004,
  lwd=1.6, xlim=c(rng_vals[1], rng_vals[2]), ylim=c(rng_vals[1], rng_vals[2]),
  err="x", pch=21, col="black", pt.bg=scales::alpha("grey70",0.7), scol="black",
  cex=2, xlab=paste0("Predicted ", paste(inVar), " (units)"),
  ylab=paste0("Observed ", paste(inVar), " (units)"),
  cex.axis=1.5, cex.lab=1.8, add=T)
legend("topleft", legend=expr, bty="n", cex=1.5)
legend("bottomright", legend=c("Prediction Interval", "Confidence Interval"),
  lty=c(1,1), col = c("grey80", "black"), lwd=3, bty="n", cex=1.5)
box(lwd=2.2)

```



```
dev.copy(png,file.path(outdir,paste0(inVar,"_PLSR_Validation_Scatterplot.png")),
         height=2800, width=3200, res=340)
```

```
## quartz_off_screen
##           3
```

```
dev.off();
```

```
## pdf
##    2
```

#### Output bootstrap results

```
# Bootstrap Coefficients
out.jk.coefs <- data.frame(Iteration=seq(1,length(bootstrap_intercept),1),
                           Intercept=bootstrap_intercept,t(bootstrap_coef))
names(out.jk.coefs) <- c("Iteration","Intercept",paste0("Wave_",wv))
head(out.jk.coefs)[1:6]
```

```
##   Iteration Intercept Wave_500 Wave_501 Wave_502 Wave_503
```



```
## 1      1  0.4731951  0.0236618987 0.021719096 0.023063691 0.02187741
## 2      2  0.5415203 -0.0007012397 0.001892634 0.008241293 0.01105366
## 3      3  0.6512533  0.0123054098 0.013428257 0.015824665 0.01772586
## 4      4 -0.9976728  0.0145306759 0.016119715 0.018834952 0.01959049
## 5      5  0.1267626  0.0076041315 0.007329090 0.009971693 0.01339406
## 6      6  0.8509641  0.0139793124 0.015195593 0.015170417 0.01434085

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

## Create core PLSR outputs

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

## [1] "Output directory:  /var/folders/tq/tydmhlwn1bdf_0pmpcq70r2c0000gn/T//RtmpAcMfZ9"
# 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: "

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

## [1] "N_g_m2_Bootstrap_PLSR_Coefficients.csv"
## [2] "N_g_m2_Bootstrap_Regression_Coefficients.png"
## [3] "N_g_m2_Cal_PLSR_Dataset.csv"
## [4] "N_g_m2_Cal_Val_Histograms.png"
## [5] "N_g_m2_Cal_Val_Scatterplots.png"
## [6] "N_g_m2_Cal_Val_Spectra.png"
```

```
## [7] "N_g_m2_Coefficient_VIP_plot.png"
## [8] "N_g_m2_Observed_PLSR_CV_Pred_15comp.csv"
## [9] "N_g_m2_PLSR_Coefficients_15comp.csv"
## [10] "N_g_m2_PLSR_Component_Selection.png"
## [11] "N_g_m2_PLSR_Validation_Scatterplot.png"
## [12] "N_g_m2_PLSR_VIPs_15comp.csv"
## [13] "N_g_m2_Val_PLSR_Dataset.csv"
## [14] "N_g_m2_Validation_PLSR_Pred_15comp.csv"
## [15] "N_g_m2_Validation_RMSEP_R2_by_Component.png"
```