Development of an intertidal foraminifera training set for the North Sea and an assessment of its application for Holocene sea-level reconstructions

Author: Dr. Graham Rush

Date Created: 2021-08-25

Copyright (c) Graham Rush, 2021

Email: G.Rush@leeds.ac.uk

The code is run in R 4.4.1

This is the R code to accompany the journal article: "Development of an intertidal foraminifera training set for the North Sea and an assessment of its application for Holocene sea-level reconstructions" by Rush et al. 2021 published in Marine Micropalentology. https://doi.org/10.1016/j.marmicro.2021.102055

The code, functions and data are provided are available to download on Figshare: https://figshare.com/authors/Graham_Rush/11546401

If you use any of this code please reference the article above and any incorporated packages.

The code is provided as a guide for carrying out an assessment of the most appropriate transfer function and training set to use. In order to assess different regions you will need to run the code multiple times. The core data provided is synthetic data and not intended for further analysis beyond this example.

If you try to run the code on your own data please contact me if you come across any problems.

This is an R Markdown Notebook. When you execute code within the notebook, the results appear beneath the code.

R code

Set up the workspace

Set up the workspace by clearing it and linking to our data files

```
rm(list = ls())
options(scipen = 6, digits = 4) # I prefer to view outputs in non-scientific notation
```

Set to the directory that you saved the download

```
# wd <- paste("Your working directory")
# setwd(wd)
```

Load up the packages we will need

```
library(rioja)
library(RColorBrewer)
library(viridis)
library(palaeoSig)
library(vegan)
library(fpc)
library(cluster)
library(dplyr)
library(plyr)
```

Load up Grahams functions into memory. These have been written by Graham Rush and incorporate functions written in the above packages. Refer to the individual functions for details.

```
setwd(paste(wd, "/functions", sep = ""))
sapply(list.files(pattern = "[.]R$", recursive = TRUE), source) # loads all the functions in the functi
setwd(wd)
```

Read in the raw data Note: the core data is not provided as part of this data

```
mod_data <- read.csv("data/northsea_forams.csv", row.names ="ID", check.names = FALSE)    # read in the
core_data <- read.csv("data/Ythan-core.csv")     # read in the core data
```

Set-up and clean the data. This creates a list with colours and names to help with the plotting and analysis later and to add the cleaned data to

```
training_sets <- list()    # create a list to store the sub-regional training sets in

## set up the colours
training_sets$cols$northsea <- viridis(9, alpha = 1) # for viridis colours
training_sets$cols$northsea <- c("#999999", "#F0E442", "#0072B2", "#E69F00", "#009E73", "#56B4E9", "#D5
training_sets$cols$west <- training_sets$cols$northsea[c(1:3, 7:9)]
training_sets$cols$northwest <- training_sets$cols$northsea[c(1,3,9)]
training_sets$cols$southwest <- training_sets$cols$northsea[c(2,7,8)]
training_sets$cols$east <- training_sets$cols$northsea[c(4:6)]
training_sets$cols$ythan <- training_sets$cols$northsea[c(9)]

## set up the labels
training_sets$names$northsea <- c("Alnmouth","Brancaster","Cowpen","Kjelst", "Rantum", "Sonderho", "Tho
training_sets$names$west <- training_sets$names$northsea[c(1:3, 7:9)]
training_sets$names$northwest <- training_sets$names$northsea[c(1,3,9)]
training_sets$names$southwest <- training_sets$names$northsea[c(2,7,8)]
training_sets$names$east <- training_sets$names$northsea[c(4:6)]
training_sets$names$ythan <- training_sets$names$northsea[c(9)]
```

Run grahams function in 'clean_regions.R' and 'cleaning.R' to clean the data 'clean_data' and remove samples with counts less than 'min.count' with the option to remove rare taxa (< 5 % in no more tha 1 % of the samples)

```
training_sets$data <- clean_regions(n = 50, rare_species = FALSE, save_file = FALSE)
```

To avoid setting up and cleaning the data each time save the data using the below code and read it in and prevent re-running the above chunk use the following code

```
save(training_sets, file="data/training_sets.RData")    # save the data
load("data/training_sets.RData")    # this line can be run to load the saved data
```

Set the workspace for each training set

Set up the data ready to analyse training sets and transfer functions. Set the region 'reg' to the region of choice. You will need to repeat the code from here on for the different regions

```
reg <- "West"     # set this to the region of choice
training.set <- training_sets$data$west    # set this to the region of choice
name <- training_sets$names$west    # set this to the region of choice

colr <- training_sets$cols$northsea
training.set$Site <- as.factor(training.set$Site)
ts.sp <- ncol(training.set)
df <- df.split(training.set, ts.sp)
env$colour <- colr[env$Site]
```

Test whether to add extra components by running a WAPLS transfer function

```
fit.wapls <- WAPLS(spec, env$SWLI)
cv.loo <- crossval(fit.wapls, cv.method = "boot", nboot = 1000)
```

Show the outcome to test whether a significant improvement occurs to justify choosing more than one component. An increase of > 5% should occur with a p value < 0.05

```
rand.t.test(cv.loo)
```

```
##         RMSE     R2 Avg.Bias Max.Bias Skill delta.RMSE     p
## Comp01 18.72 0.6329 -0.03947    62.83 63.29    -38.573 0.001
## Comp02 17.73 0.6853  0.19955    62.34 68.41     -5.245 0.009
## Comp03 17.37 0.7306  0.35585    53.69 72.88     -2.064 0.020
## Comp04 17.82 0.7372  0.47816    48.31 73.41      2.595 0.358
## Comp05 19.02 0.7344  0.86683    43.76 72.67      6.753 0.774
```

Adjust the number of desired components and create a label for later

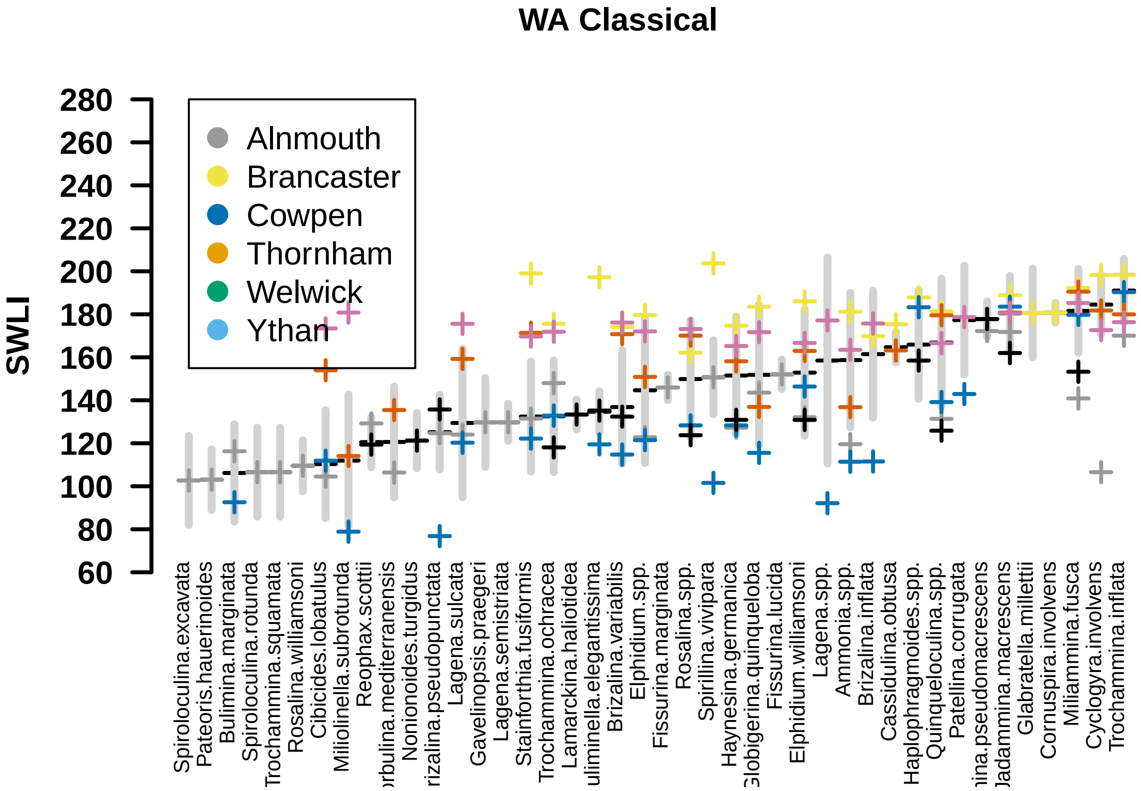
```
wapls.mod <- 2     # adjust this to fit the best choice of components
main1 = paste("WAPLS Component", wapls.mod, " ")
```

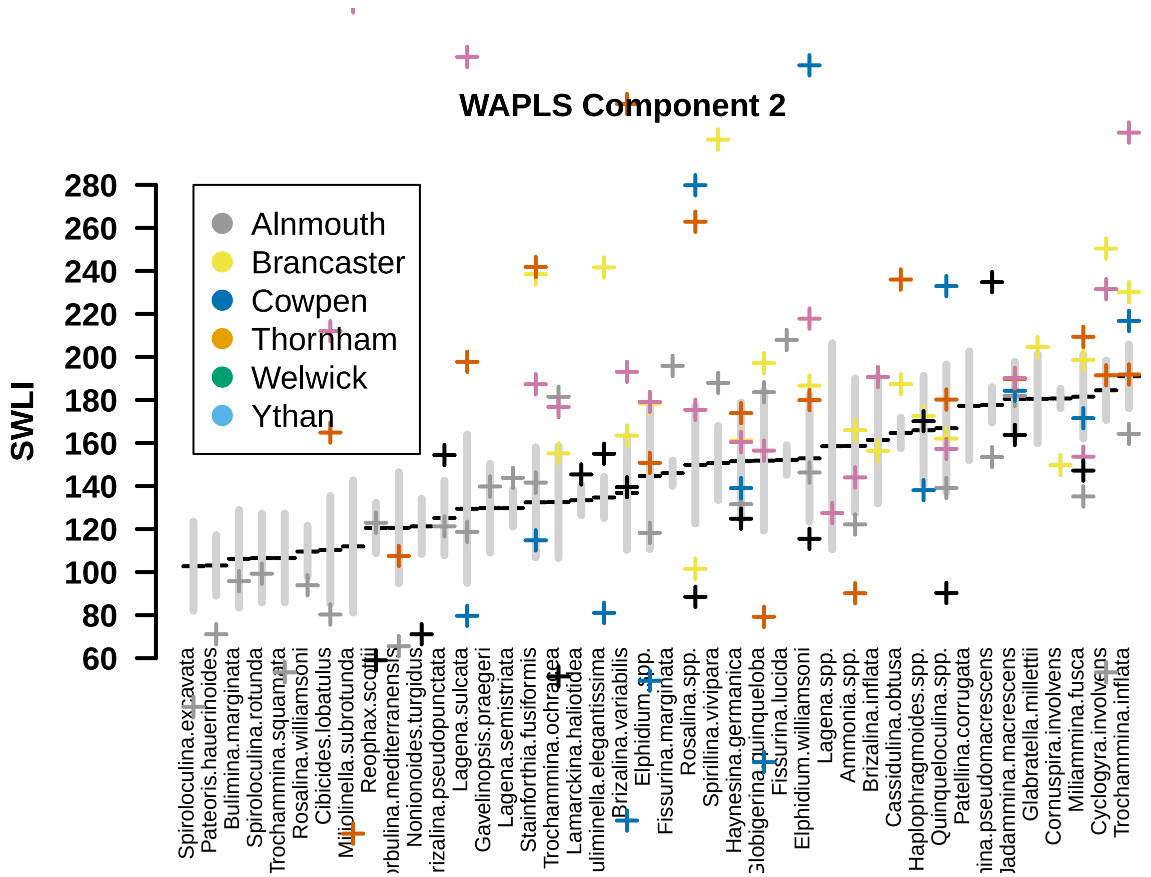
Analysis of training set

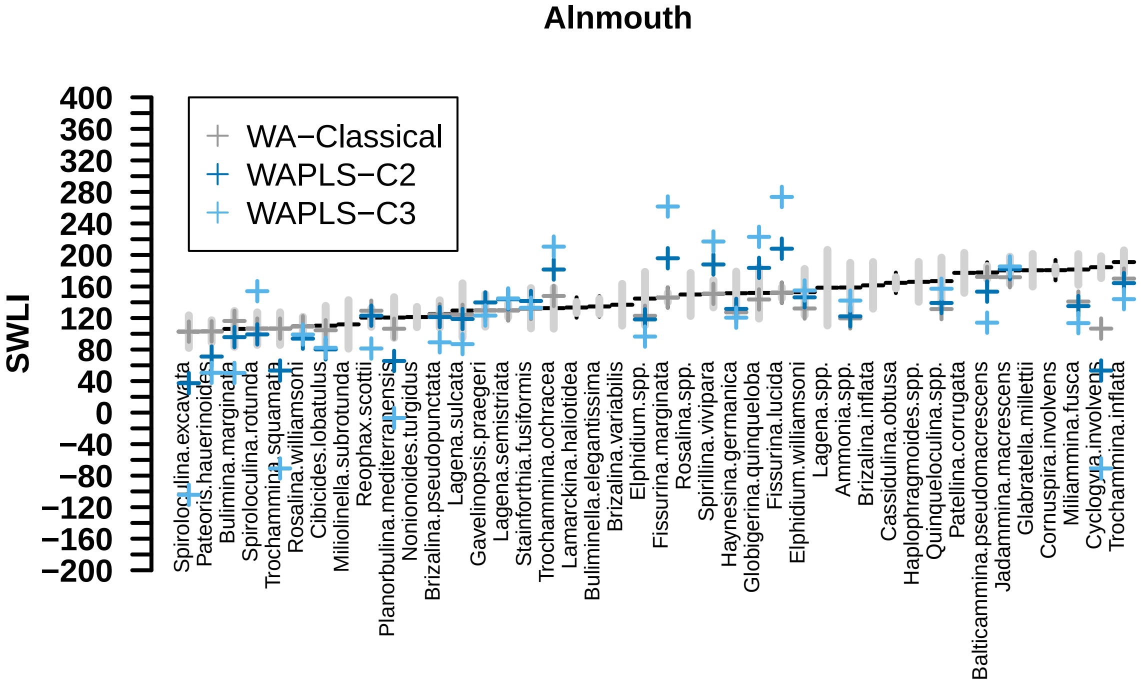
Plot species optimas

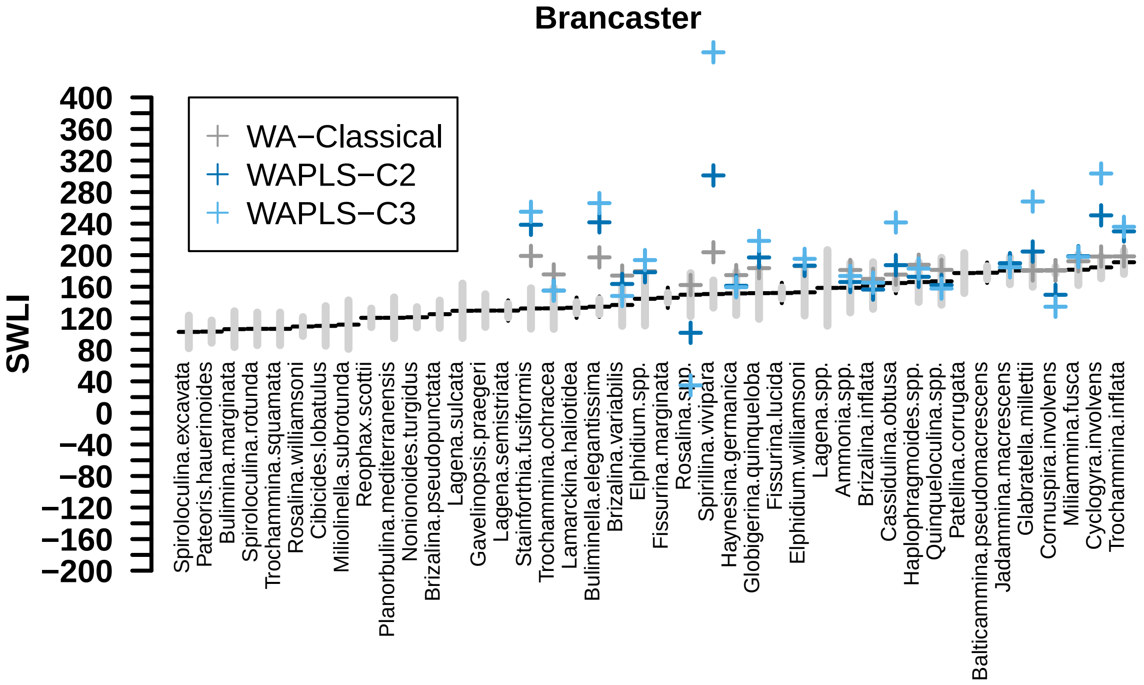
Use grahams functions in plot_optimas to plot the species tolerances

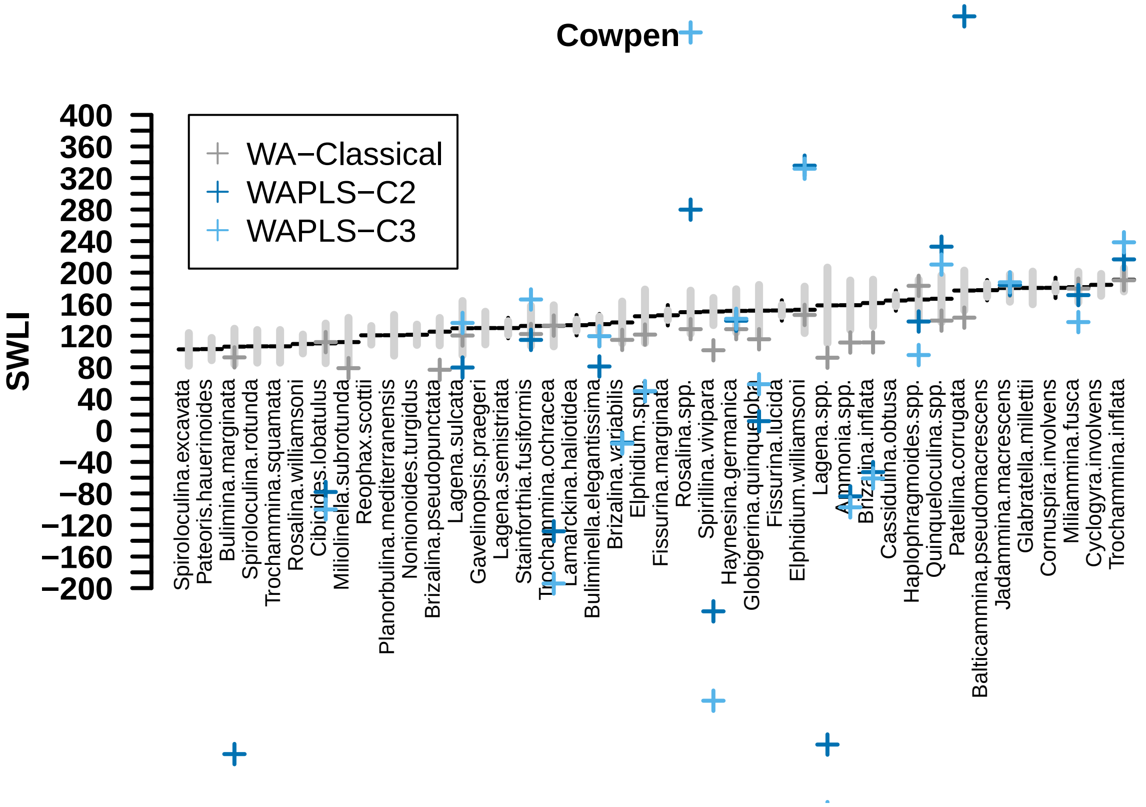
```
plot_species_tolerances(model = 2, y_0 = 60, y_1 = 280, x_1 = ncol(spec), wa = TRUE, wapls = TRUE, compo
```


WA Classical
SWLI
280
260
240
220
200
180
160
140
120
100
80
60
● Alnmouth
● Brancaster
● Cowpen
● Thornham
● Welwick
● Ythan
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
rbulina.mediterranensis
Nonionoides.turgidus
rizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
uliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
ina.pseudomacrescens
ladammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata

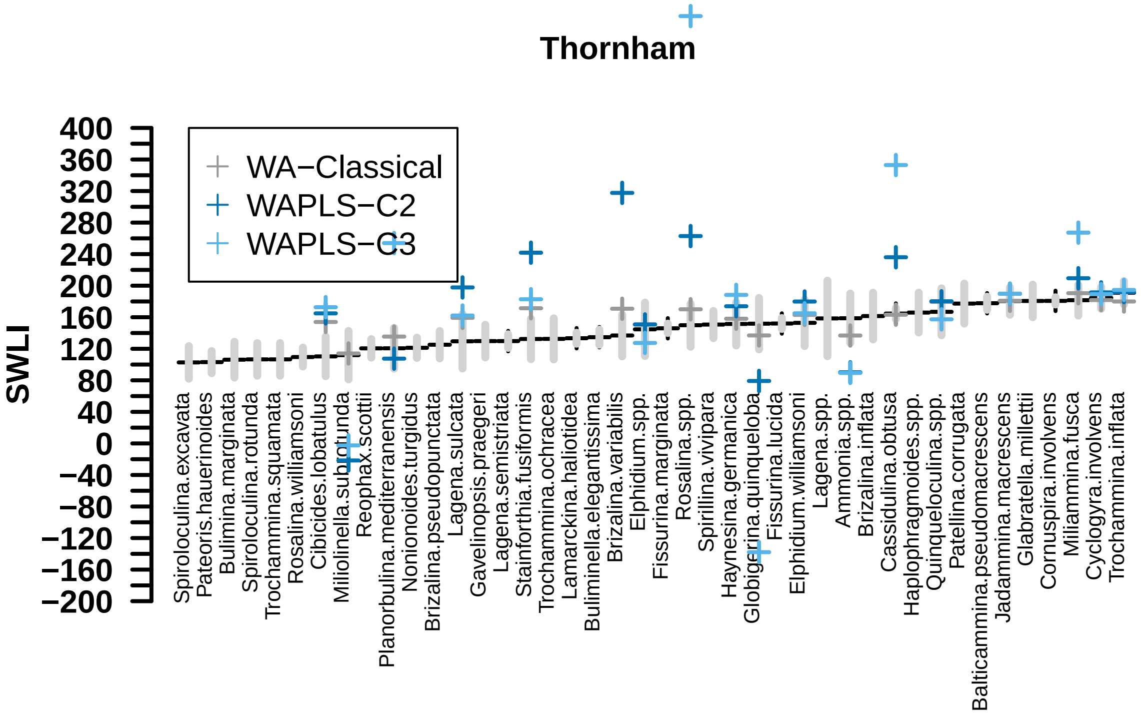

WAPLS Component 2
SWLI
280
260
240
220
200
180
160
140
120
100
80
60
● Alnmouth
● Brancaster
● Cowpen
● Thornham
● Welwick
● Ythan
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
›rbulina.mediterranensis
Nonionoides.turgidus
rizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
uliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
ina.pseudomacrescens
Jadammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata

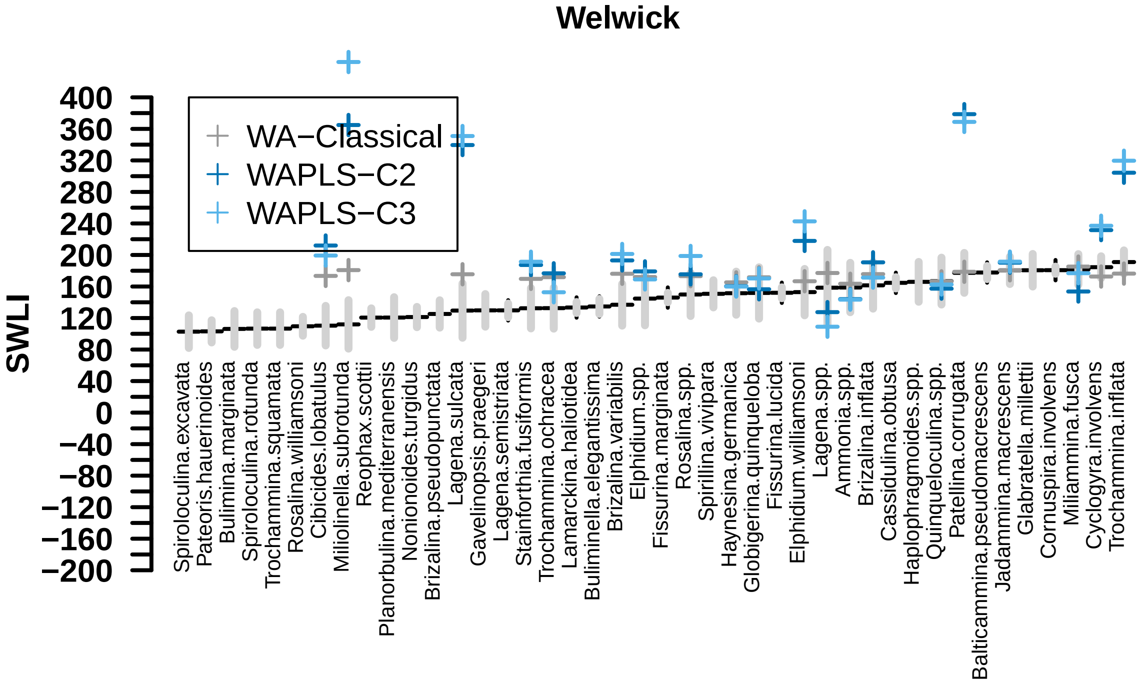

Alnmouth
WA−Classical
WAPLS−C2
WAPLS−C3
SWLI
400
360
320
280
240
200
160
120
80
40
0
−40
−80
−120
−160
−200
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
Planorbulina.mediterranensis
Nonionoides.turgidus
Brizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
Buliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
Balticammina.pseudomacrescens
Jadammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata

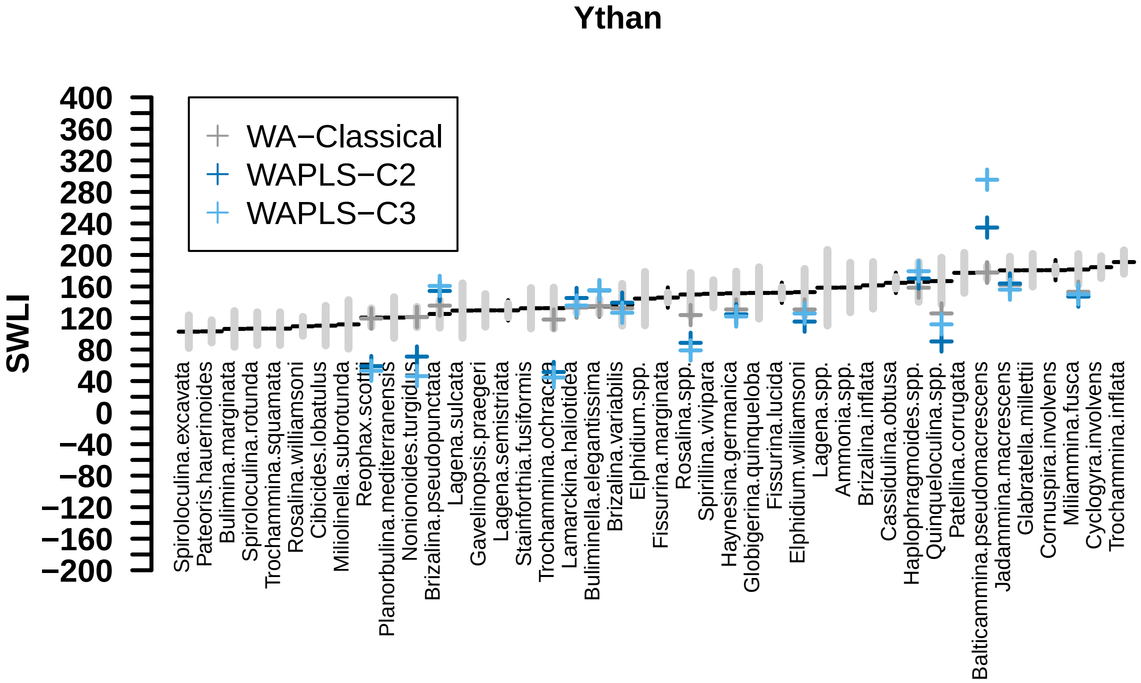

Brancaster
SWLI
400
360
320
280
240
200
160
120
80
40
0
−40
−80
−120
−160
−200
WA−Classical
WAPLS−C2
WAPLS−C3
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
Planorbulina.mediterranensis
Nonionoides.turgidus
Brizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
Buliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
Balticammina.pseudomacrescens
Jadammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata



Cowpen
SWLI
400
360
320
280
240
200
160
120
80
40
0
−40
−80
−120
−160
−200
WA−Classical
WAPLS−C2
WAPLS−C3
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
Planorbulina.mediterranensis
Nonionoides.turgidus
Brizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
Buliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
Balticammina.pseudomacrescens
Jadammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata


Thornham
WA−Classical
WAPLS−C2
WAPLS−C3
SWLI
400
360
320
280
240
200
160
120
80
40
0
−40
−80
−120
−160
−200
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
Planorbulina.mediterranensis
Nonionoides.turgidus
Brizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
Buliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
Balticammina.pseudomacrescens
Jadammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata


Welwick
SWLI
400
360
320
280
240
200
160
120
80
40
0
−40
−80
−120
−160
−200
WA−Classical
WAPLS−C2
WAPLS−C3
Spiroloculina.excavata
Pateoris.hauerinoides
Bulimina.marginata
Spiroloculina.rotunda
Trochammina.squamata
Rosalina.williamsoni
Cibicides.lobatulus
Miliolinella.subrotunda
Reophax.scottii
Planorbulina.mediterranensis
Nonionoides.turgidus
Brizalina.pseudopunctata
Lagena.sulcata
Gavelinopsis.praegeri
Lagena.semistriata
Stainforthia.fusiformis
Trochammina.ochracea
Lamarckina.haliotidea
Buliminella.elegantissima
Brizalina.variabilis
Elphidium.spp.
Fissurina.marginata
Rosalina.spp.
Spirillina.vivipara
Haynesina.germanica
Globigerina.quinqueloba
Fissurina.lucida
Elphidium.williamsoni
Lagena.spp.
Ammonia.spp.
Brizalina.inflata
Cassidulina.obtusa
Haplophragmoides.spp.
Quinqueloculina.spp.
Patellina.corrugata
Balticammina.pseudomacrescens
Jadammina.macrescens
Glabratella.millettii
Cornuspira.involvens
Miliammina.fusca
Cyclogyra.involvens
Trochammina.inflata



Cluster Analysis

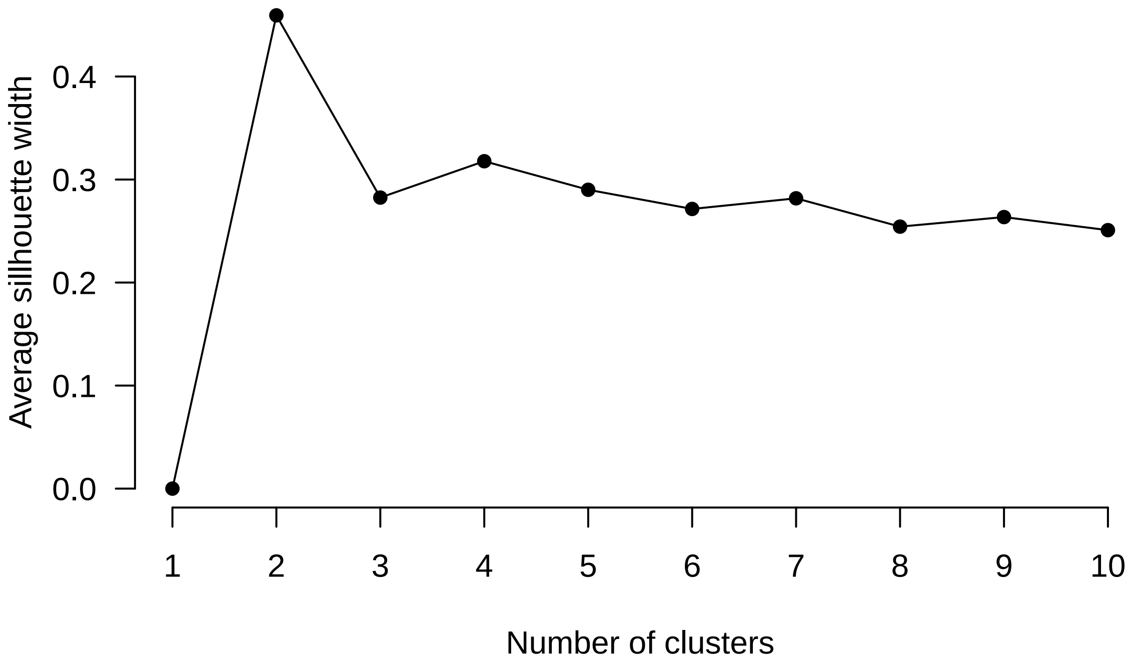
Run the cluster analysis

```
df.split (training.set, ts.sp)    # split the data
dis <- vegdist(spec, "euclidean")    #  set distance based on rule. Set this e.g. "bray" is bray curtis
clusters <- pamk(dis, krange = 1:10, criterion = "multiasw", critout = TRUE, metric = "euclidean")    #
```

```
## 1  clusters  0
## 2  clusters  0.4593
## 3  clusters  0.2825
## 4  clusters  0.3178
## 5  clusters  0.29
## 6  clusters  0.2714
## 7  clusters  0.2818
## 8  clusters  0.2543
## 9  clusters  0.2636
## 10  clusters  0.2509
```

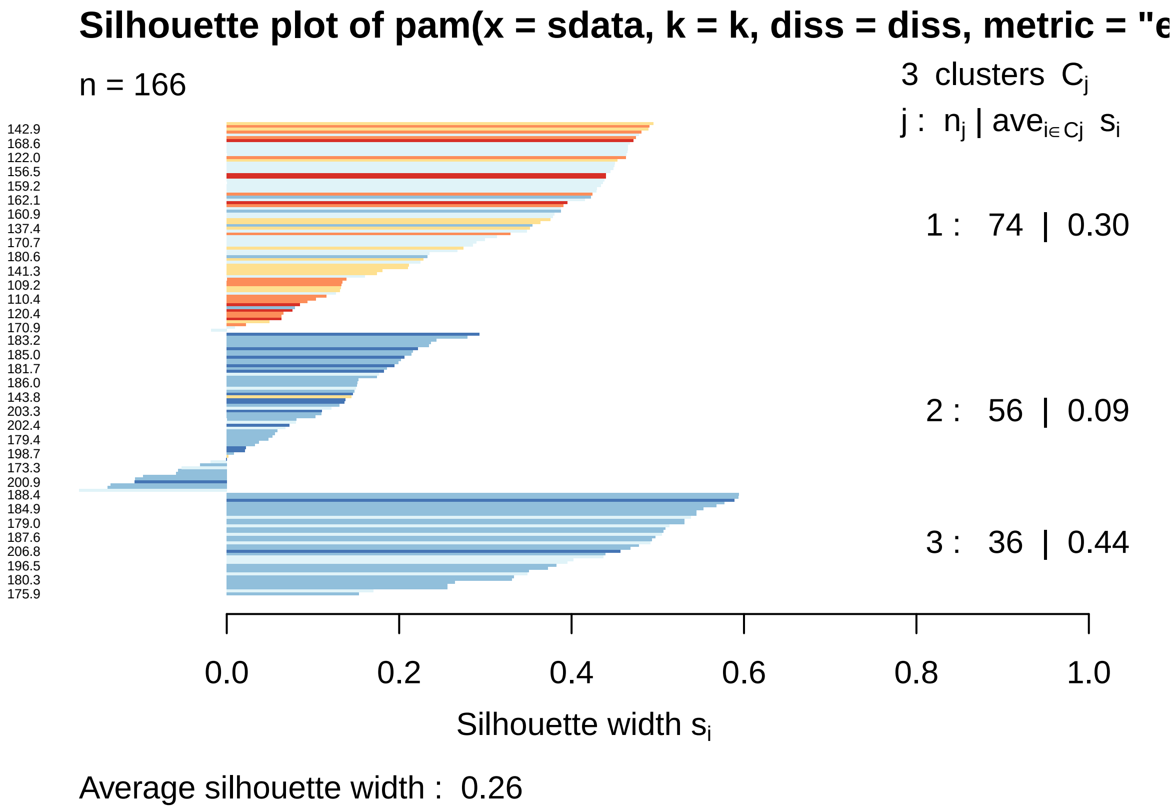
Plot the clusters in a broken hockey stick using grahams function ‘pam_hockey_stick.R’

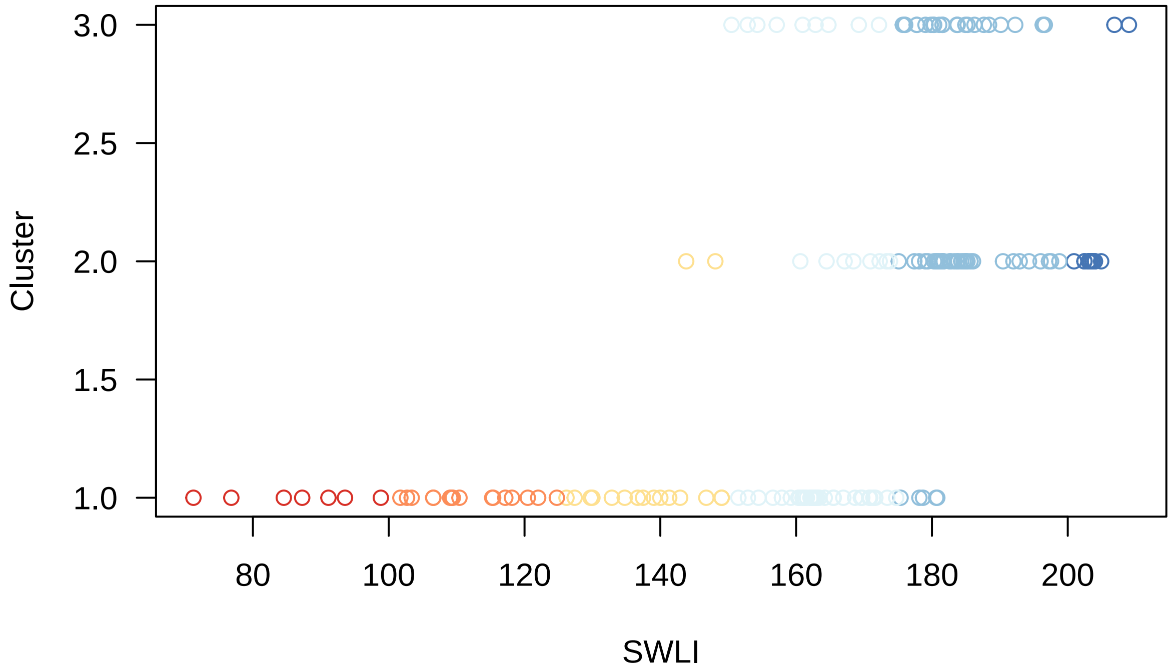
```
pam_hockey_stick()     # uses grahams functions in pam_hockey_stick to plot broken stick graph to asses
```

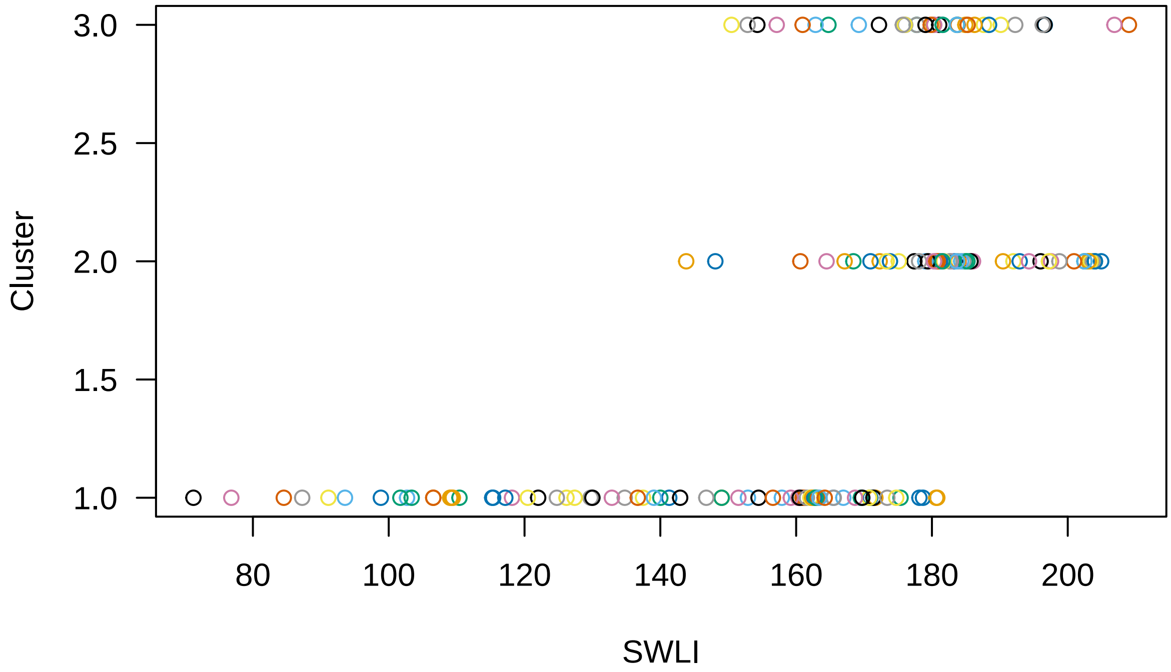


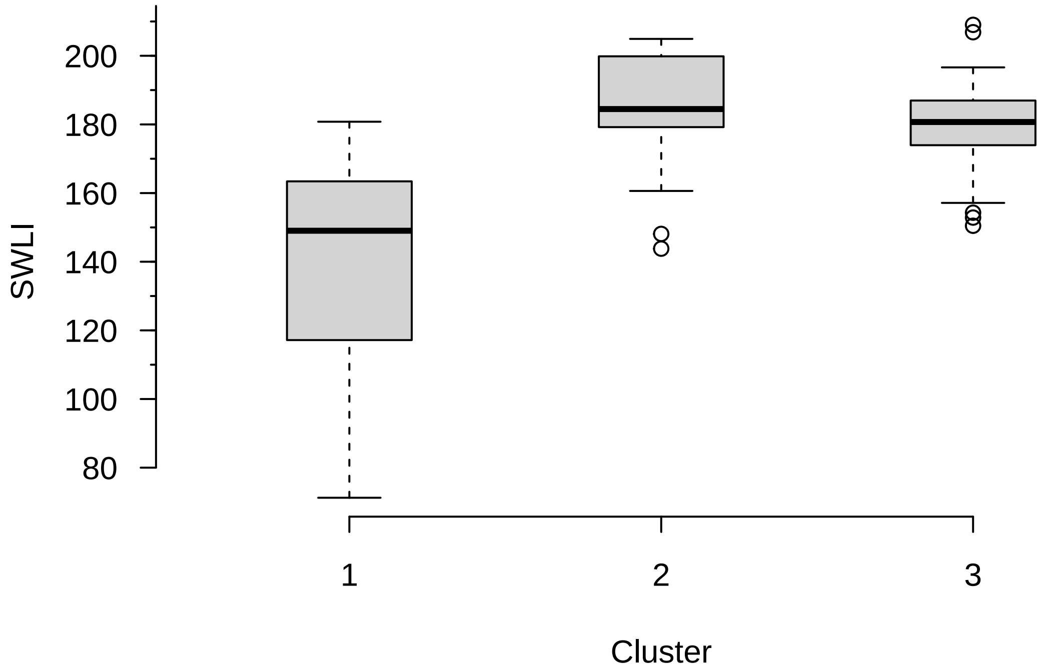
Set to the desired number of clusters and plot the clusters using grahams functions in 'plot_pam.R'. Four plots are produde: 1. A silhouette plot, 2. The samples goruped by cluster, plotted against SWLI and coloured by SWLI, 3. The samples goruped by cluster, plotted against SWLI and coloured by site. 4. Boxplot of the clusters against SWLI.

```
cluster <- pamk(dis, krange = 3, criterion = "multiasw", critout = TRUE, metric = "euclidean")     # se
pt <- plot.pamk(k = 3, p2 = TRUE, p3 = TRUE, boxplot = TRUE)    # plot the clusters
```


Silhouette plot of pam(x = sdata, k = k, diss = diss, metric = "e
n = 166
3 clusters C_j
j : n_j | $ave_{i\in Cj}$ s_i
1 : 74 | 0.30
2 : 56 | 0.09
3 : 36 | 0.44
142.9
168.6
122.0
156.5
159.2
162.1
160.9
137.4
170.7
180.6
141.3
109.2
110.4
120.4
170.9
183.2
185.0
181.7
186.0
143.8
203.3
202.4
179.4
198.7
173.3
200.9
188.4
184.9
179.0
187.6
206.8
196.5
180.3
175.9
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.26


3.0
2.5
2.0
1.5
1.0
Cluster
80
100
120
140
160
180
200
SWLI


3.0
2.5
2.0
1.5
1.0
Cluster
80
100
120
140
160
180
200
SWLI



Create a table of the data

```
clustering <- as.data.frame(pt)
clustering
```

```
##   cluster size av.widths max_diss av_diss med_swli min_swli max_swli
## 1       1   74      0.30    78.43   35.31    93.56    71.24    180.8
## 2       2   56      0.09   105.33   33.18   203.08   143.81    204.9
## 3       3   36      0.44    44.61   20.90   181.55   150.48    209.0
```

CCA & DCA

Run correlation analysis to understand the relationships.

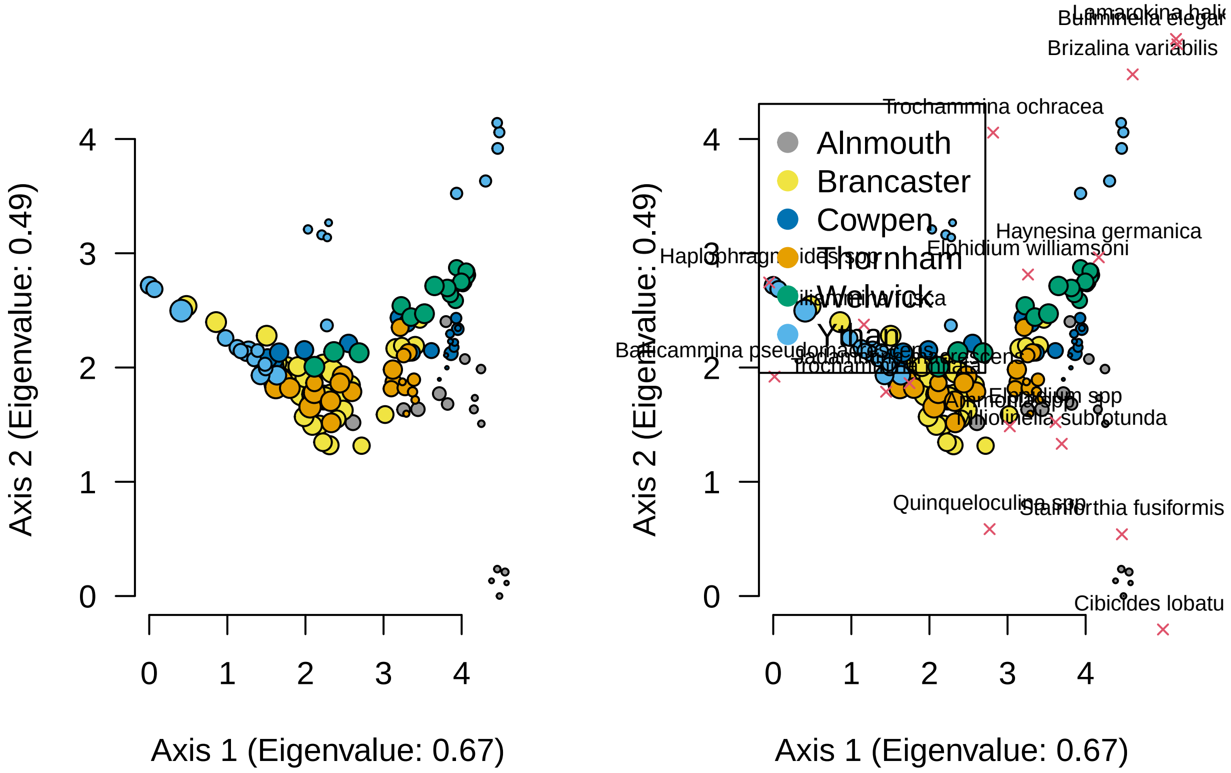
```
pcca <- cca(spec, env$SWLI)    # run CCA
w.cca <- round((pcca$CCA$eig / pcca$tot.chi) * 100, 2)    # extract the proportion of variance explained
w.cca    # show the value
```

```
##  CCA1
## 10.73
```

```
ns.dca <- decorana(veg=spec)    # run dca
ns.dca
```

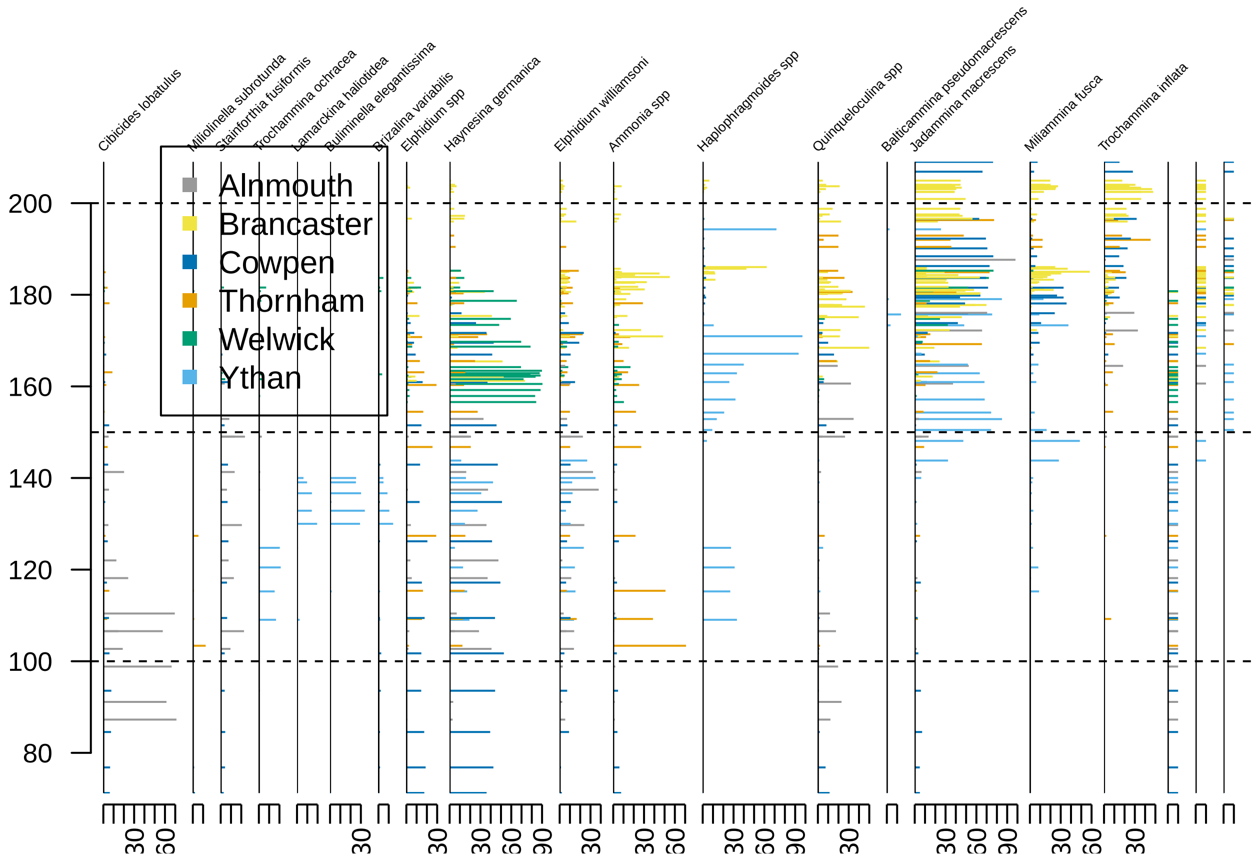
```
##
## Call:
## decorana(veg = spec)
##
## Detrended correspondence analysis with 26 segments.
## Rescaling of axes with 4 iterations.
##
##                   DCA1  DCA2  DCA3  DCA4
## Eigenvalues      0.663 0.421 0.239 0.209
## Decorana values  0.670 0.488 0.218 0.127
## Axis lengths     4.575 4.141 2.747 1.821
```

```
plot.new()
plot_dca(origin = T)
```



plot the foram species diagram using grahams function in ‘plot_foram_diagram.R’

```
plot_foram_diagram(data = training.set, min.species = 10, clusters = 3, method = "euclidean", plot_zone
```



Perform the transfer functions

WA

```
df.split (training.set, ts.sp)     # split the data
env$colour <- training_sets$cols$northsea[env$Site]

fit.wa <- WA(spec,env$SWLI, tolDW = T)     # run the WA transfer function
cv <- crossval(fit.wa, cv.method = "boot", nboot = 1000)    # perform cross-validation using bootstrappi
cv.wa.loso <- crossval(fit.wa, cv.method = "lgo", ngroups = env$Site)     # perform cross-validation usi
test.wa <- as.data.frame(rand.t.test(cv))
```

```
test.wa    # show results
```

```
##              RMSE     R2  Avg.Bias Max.Bias Skill delta.RMSE  p
## WA.inv      18.74 0.6324 -0.028159    62.68 63.24         NA NA
## WA.cla      22.66 0.6344  0.007916    50.72 47.25         NA NA
## WA.inv.tol 17.78 0.6745 -0.340427    64.37 67.42     -5.120  0
## WA.cla.tol 20.73 0.6757 -0.402279    55.15 56.85     -8.537  0
```

WAPLS

```
fit.wapls <- WAPLS(spec,env$SWLI)    # run the WAPLS transfer function
cv.loo <- crossval(fit.wapls, cv.method= "boot", nboot = 1000)    # perform cross-validation using boots
cv.loso <- crossval(fit.wapls, cv.method= "lgo", ngroups = env$Site)     # perform cross-validation usin
test.wapls.loo <- as.data.frame(rand.t.test(cv.loo))
test.wapls.loso <- as.data.frame(rand.t.test(cv.loso))
```

```
test.wapls.loo    # show results
```

```
##         RMSE     R2 Avg.Bias Max.Bias Skill delta.RMSE     p
## Comp01 18.74 0.6323 -0.03585    62.60 63.23    -38.489 0.001
## Comp02 17.79 0.6844  0.20010    62.00 68.32     -5.099 0.020
## Comp03 17.32 0.7309  0.32536    53.48 72.92     -2.605 0.018
## Comp04 17.65 0.7387  0.41108    47.78 73.60      1.915 0.291
## Comp05 18.52 0.7359  0.82169    43.04 72.86      4.886 0.758
```

LW

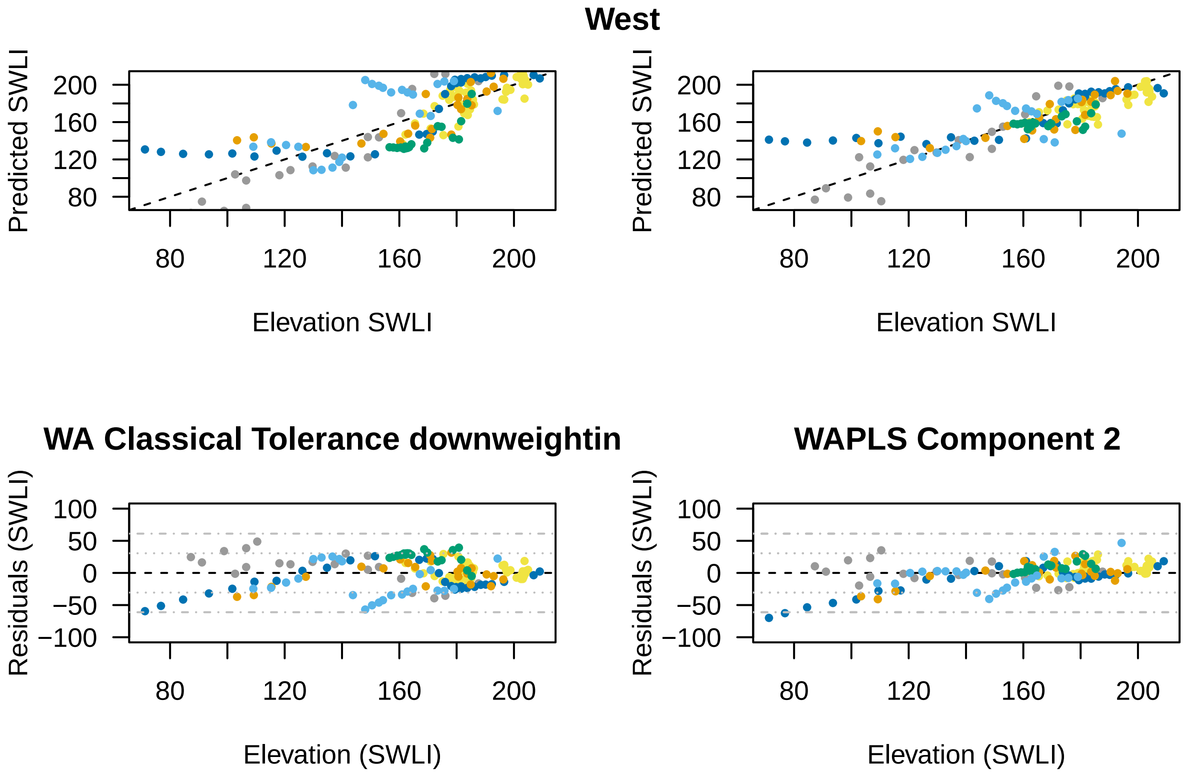
```
fit.lw <- LWR(spec, env$SWLI, FUN=WAPLS, dist.method="sq.chord", k=50)     # run the locally weighted tr
cv.lw <- crossval(fit.lw, cv.method="boot", nboot = 100)     # cross validate the results using bootstra
```

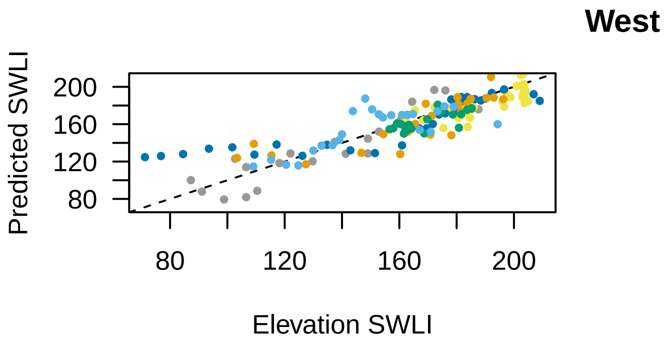
```
cv.lw     # show the results
```

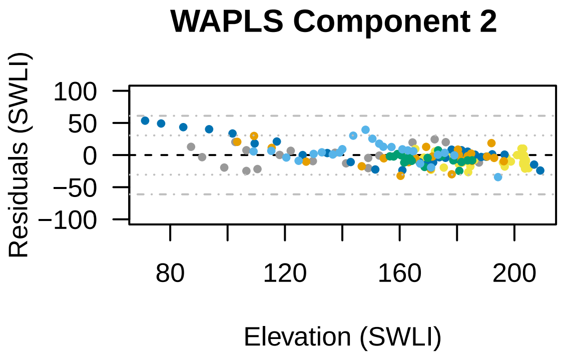
```
##
## Method : Locally Weighted Regression
## Call   : LWR(y = y, x = x, FUN = WAPLS, dist.method = "sq.chord", k = 50,
##     lean = TRUE)
## Distance           : sq.chord
## No. samples        : 166
## No. species        : 42
## No. local          : 50
##
## Performance:
##                RMSE      R2  Avg.Bias  Max.Bias  Skill
## Comp01        16.16  0.7201    0.3628     56.27  71.87
## Comp02        13.70  0.8013    1.5608     43.15  79.77
## Comp03        13.64  0.8011    1.1919     37.77  79.95
## Comp04        14.36  0.7790    0.6328     37.41  77.78
## Comp05        16.13  0.7297    0.6771     39.03  71.97
## Comp01_XVal  17.63  0.6674    1.4233     60.28  66.51
## Comp02_XVal  14.77  0.7680    1.6195     48.69  76.52
## Comp03_XVal  13.77  0.7966    0.6810     37.83  79.58
## Comp04_XVal  14.02  0.7894    0.7923     34.97  78.83
## Comp05_XVal  14.37  0.7786   -0.0464     33.43  77.77
```

Plot the residuals and predictions against the true value with 1 and 2 standard deviations shown using grahams functions in ‘plot_cross_validation.R’

```
plot_residuals(wa.mod = 2, zoom = TRUE)     # set to the desired wa deshrinking method. 2 = classical
plot_residuals_lw(model = 2, zoom = TRUE)
```


West
Predicted SWLI
200
160
120
80
80
120
160
200
Elevation SWLI
Predicted SWLI
200
160
120
80
80
120
160
200
Elevation SWLI
WA Classical Tolerance downweightin
Residuals (SWLI)
100
50
0
−50
−100
80
120
160
200
Elevation (SWLI)
WAPLS Component 2
Residuals (SWLI)
100
50
0
−50
−100
80
120
160
200
Elevation (SWLI)





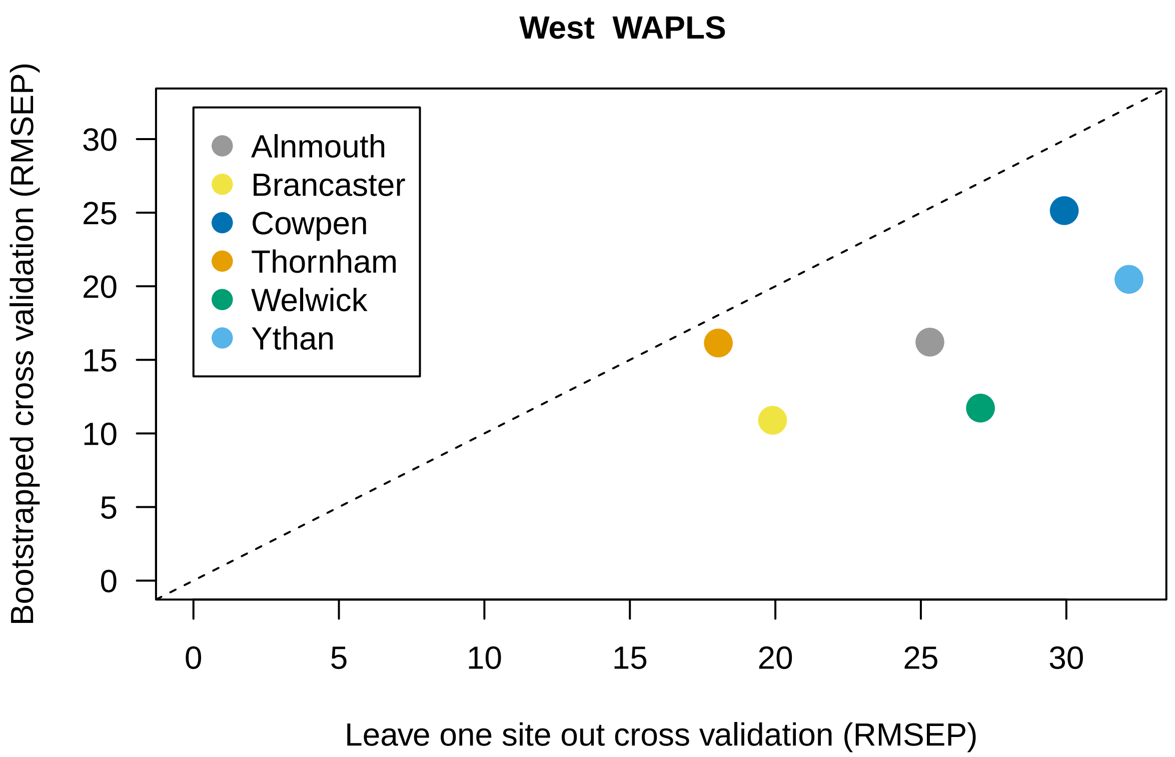
Compare cross-validation using leave-one-site-out using Grahams function in ‘loso.R’

```
compare <- loso(wapls.mod, name)
```

```
##                LOO  LOSO       Site
## Alnmouth    16.20 25.31   Alnmouth
## Brancaster 10.89 19.90 Brancaster
## Cowpen      25.14 29.93     Cowpen
## Thornham    16.15 18.04   Thornham
## Welwick     11.72 27.05    Welwick
## Ythan       20.47 32.15      Ythan
```

Plot the bootstrapping and loso cross-validation

```
plot_loso (zoom = TRUE, text = FALSE)
```



Save the results using Grahams function ‘saver.R’

```
results <- saved(region = reg, method = paste("WAPLS-C", wapls.mod))
file_name <- paste("results/", "tf_", reg, ".csv", sep = "")
write.csv(results, file_name)
file_name <- paste("results/", "WAPLS-C", wapls.mod, "-compare-", reg, ".csv", sep = "")
write.csv(compare, file = file_name)
```

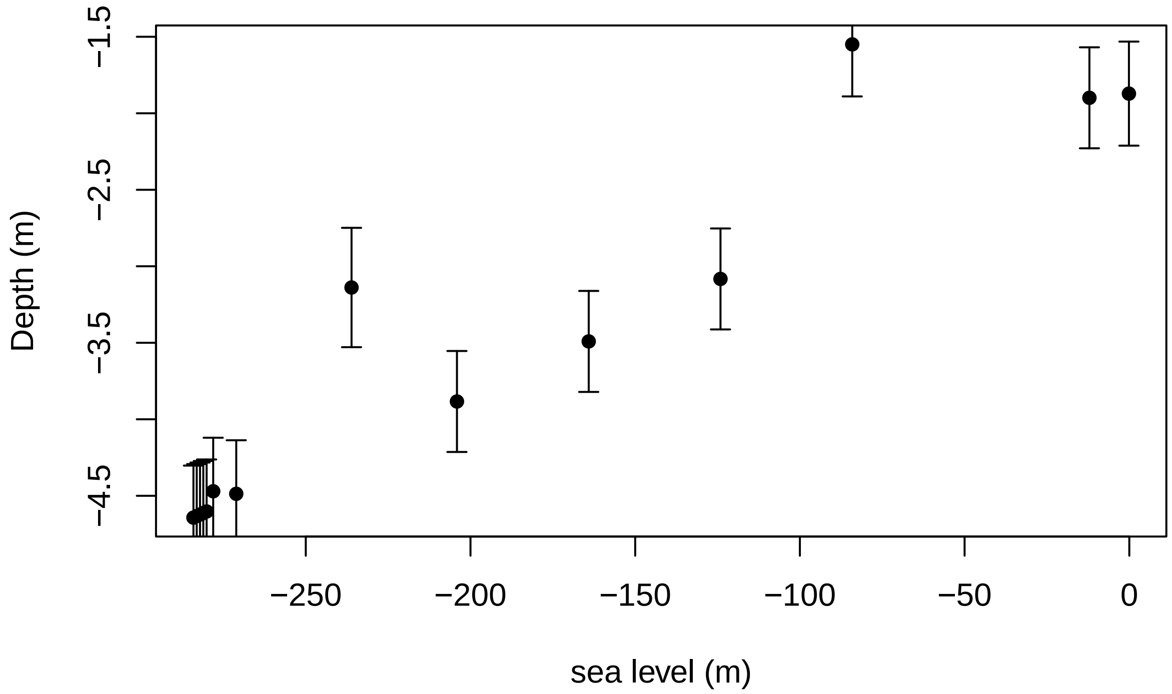
Perform the reconstructions

Clean the core using grahams function in ‘cleaning.R’

```
core_clean <- clean_core(core = core_data, s = 15, td = 1, rare = F, f = F)
core <- c.split (core_clean, s = 9) # split the core to species and environmental data.  15 is the last
```

Predict the SWLI for core samples using WA using grahams function in ‘Reconstructions.R’ Change the components to match the deshrinking method

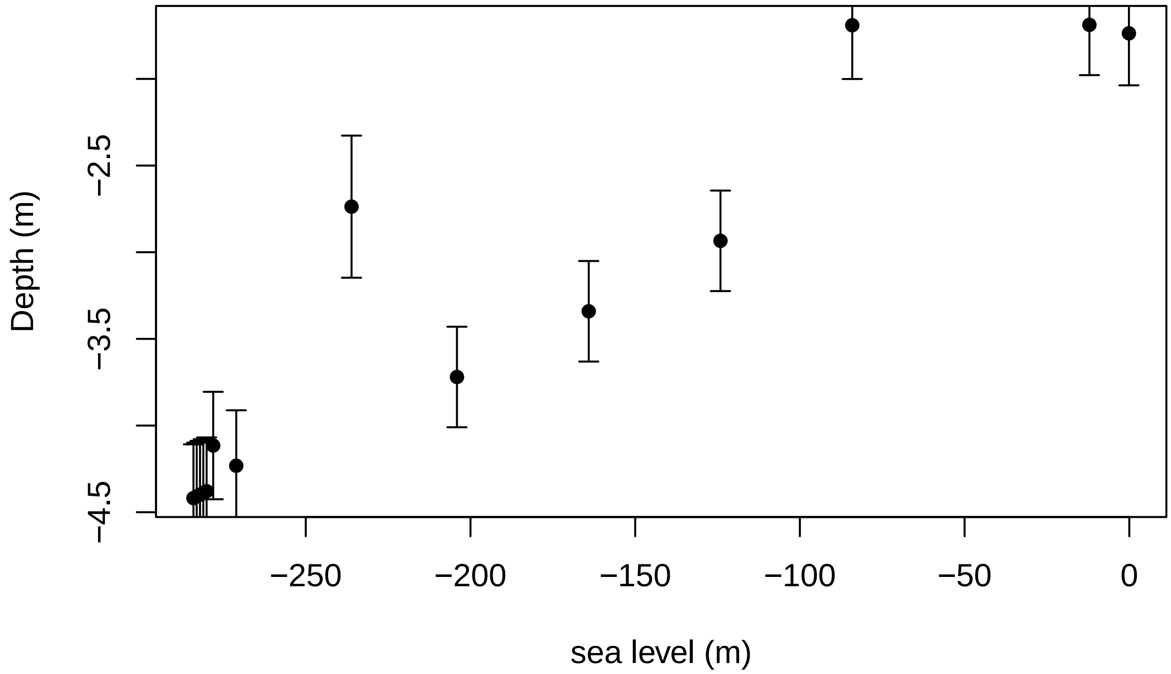
```
prediction_wa <- reconstruct_core(core = core, tf = fit.wa, components = 4, species = 9, mhhw = 1.95, m
```



```
core_clean <- cbind(core_clean, prediction_wa)    # add to the data
```

Predict the SWLI for core samples using WAPLS using grahams function in 'Reconstructions.R' Change the components to match the components

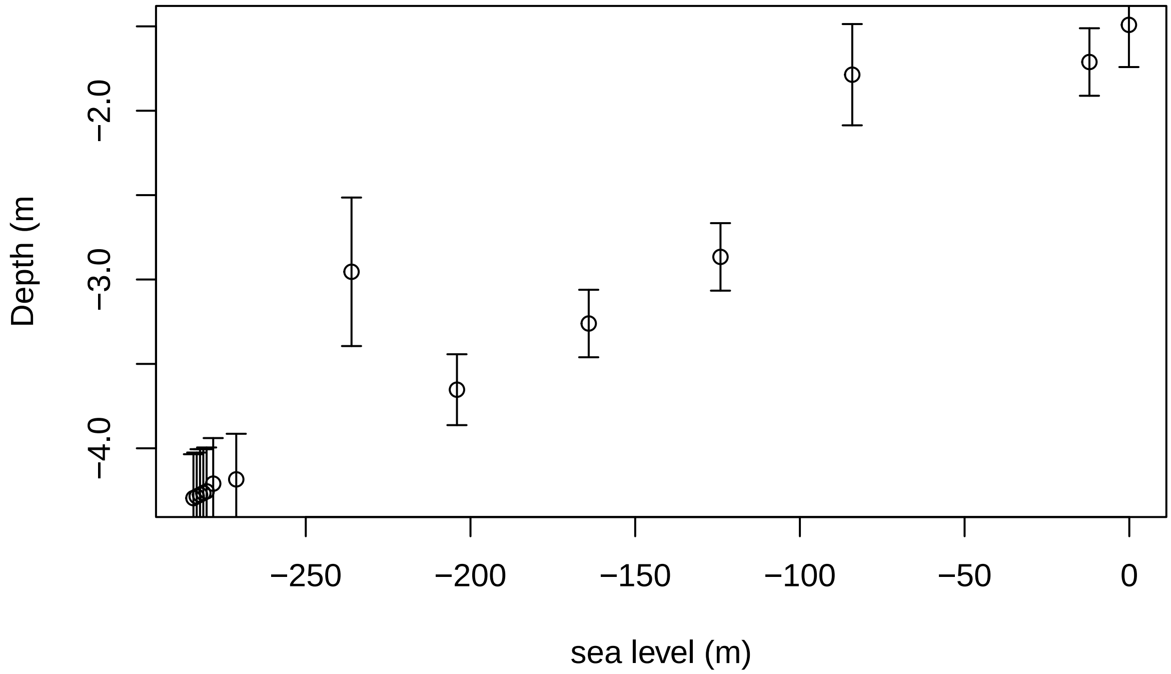
```
prediction_wapls <- reconstruct_core(core = core, tf = fit.wapls, components = 2, species = 9, mhhw = 1
```



```
core_clean <- cbind(core_clean, prediction_wapls)     # add to data for exporting and comparison
```

Predict the SWLI for core samples using Locally weighted - WAPLS

```
prediction_lw <- run_lw_tf(n_analogues = 30, components = 2, tf = WAPLS, species = 9, mhhw = 1.95, mtl
```

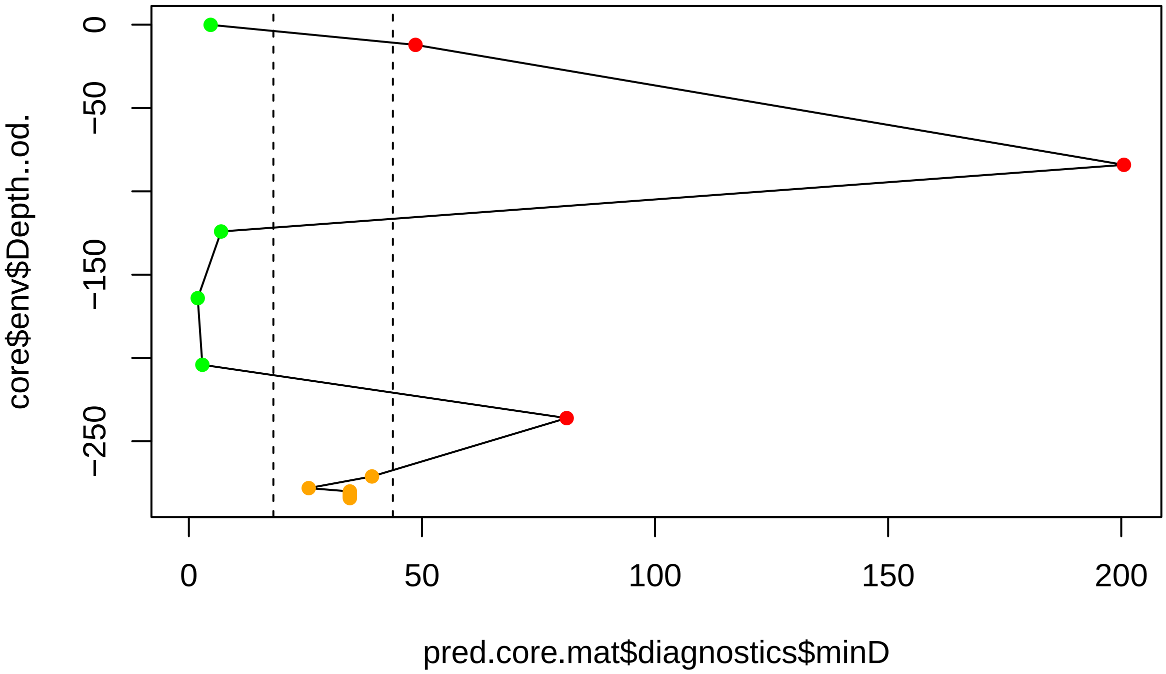


```
core_clean <- cbind(core_clean, prediction_lw$sea_level)     # add to the data -->
```

–> –>

Test the analogues. using MAT or goodness of fit using grahams functions in 'test-mat.R'

```
mat <- run.mat(z = 50, d = "sq.chord", plot = TRUE)     # run the mat.
```



```
core_clean <- cbind(core_clean, mat$mat)     # add to the data
# gof <- goodness_of_fit(plot = TRUE)     # run the goodness of fit
# core_clean <- cbind(core_clean, gof)     # add to the data
```

Save the results for each region

```
file_name <- paste("results/", "reconstructions_", reg, ".csv", sep = "")
write.csv(core_clean, file_name)
```

Test whether the reconstructions are significant

```
ralgh <- randomTF(spp = spec, env = env$SWLI, fos = core$spec, n = 99, fun = WAPLS, col = 2)
ralgh.wa <- randomTF(spp = spec, env = env$SWLI, fos = core$spec, n = 99, fun = WA, col = 2)
ralgh$sig     # show the data
```

```
##  env
## 0.77
```

```
ralgh.wa$sig     # show the data
```

```
##  env
## 0.72
```