

Complete-case analysis

Luciano L.M. De Benedictis

2026-01-08

```
# setup -----
library(tidyverse)

## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.6
## v forcats   1.0.1     v stringr   1.6.0
## v ggplot2   4.0.1     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr    1.3.2
## v purrr    1.2.0
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(GET)
library(vegan)

## Loading required package: permute

library(parallel)
library(patchwork)

resampled <- readRDS("transect_resampled.rds")
speciestraits <- readRDS("traits_noimp.rds")
source("0.functions.R")

indices <- vector(mode = "list")
indexcurves <- vector(mode = "list")

#update boxcox to support NAs
boxcox <- function(x, check = F) {

  x1 <- na.omit(x)

  if (any(x1 == 0)) x1 <- x1 + 1

  boxcox_result <- MASS::boxcox(x1 ~ 1, plotit = F)
  lambda <- boxcox_result$x[which.max(boxcox_result$y)]
```

```

message("Box-Cox transform with lambda = ", lambda)

if (check == T) return(lambda)
else{
  if (lambda == 0) {
    x1 <- log(x1)
  } else {
    x1 <- (x1 ^ lambda - 1) / lambda
  }
  x[!is.na(x)] <- x1
  return(x)
}

# keep complete traits ----

remove_overl <- speciestraits |>
  filter(if_any(leaf_area_mm2:spread, is.na)) |>
  pull(species)

remove_above <- speciestraits |>
  filter(if_any(leaf_area_mm2:ssd_combined_mg_mm3, is.na)) |>
  pull(species)

remove_clo <- speciestraits |>
  filter(if_any(BBRsize:spread, is.na)) |>
  pull(species)

resampled_overl <- resampled |>
  lapply(function(x) lapply(x, function(y) filter(y, !species %in% remove_overl)))

resampled_above <- resampled |>
  lapply(function(x) lapply(x, function(y) filter(y, !species %in% remove_above)))

resampled_clo <- resampled |>
  lapply(function(x) lapply(x, function(y) filter(y, !species %in% remove_clo)))

rm(resampled)

# transform traits ----

speciestraits |>
  select(-n) |>
  summarize(across(where(is.numeric), ~ boxcox(., check = T))) |>
  pivot_longer(everything(), names_to = "trait", values_to = "lambda") |>
  arrange(lambda, trait)

## Box-Cox transform with lambda = 0
## Box-Cox transform with lambda = -0.6
## Box-Cox transform with lambda = -0.4
## Box-Cox transform with lambda = 0.3
## Box-Cox transform with lambda = 0.2
## Box-Cox transform with lambda = 0

```

```

## Box-Cox transform with lambda = 0.8
## Box-Cox transform with lambda = 1.7
## Box-Cox transform with lambda = -0.2
## Box-Cox transform with lambda = -0.0999999999999999

## # A tibble: 10 x 2
##   trait           lambda
##   <chr>          <dbl>
## 1 nmass_mg_g    -0.6
## 2 lma_g_m2      -0.4
## 3 offspring      -0.2
## 4 spread        -0.1000
## 5 leaf_area_mm2 0
## 6 ssd_combined_mg_mm3 0
## 7 diaspore_mass_mg 0.200
## 8 plant_height_m 0.300
## 9 BBRsize        0.8
## 10 persistence   1.7

```

This is a dry run check. The traits will be transformed as such:

- $\sim 1/\sqrt{N}$ mass, LMA, offspring
- $\sim \log$ spread, SSD, leaf area, diaspore mass
- $\sim \sqrt{\text{height}}$
- $\sim \text{identity}$ BBR size
- $\sim \text{square}$ persistence

```

data <- speciestraits |>
  select(species, where(is.numeric)) |>
  select(-n) |>
  mutate(across(where(is.numeric), ~ boxcox(.))) |>
  as.data.frame()

```

```

## Box-Cox transform with lambda = 0

## Box-Cox transform with lambda = -0.6

## Box-Cox transform with lambda = -0.4

## Box-Cox transform with lambda = 0.3

## Box-Cox transform with lambda = 0.2

## Box-Cox transform with lambda = 0

## Box-Cox transform with lambda = 0.8

## Box-Cox transform with lambda = 1.7

## Box-Cox transform with lambda = -0.2

## Box-Cox transform with lambda = -0.0999999999999999

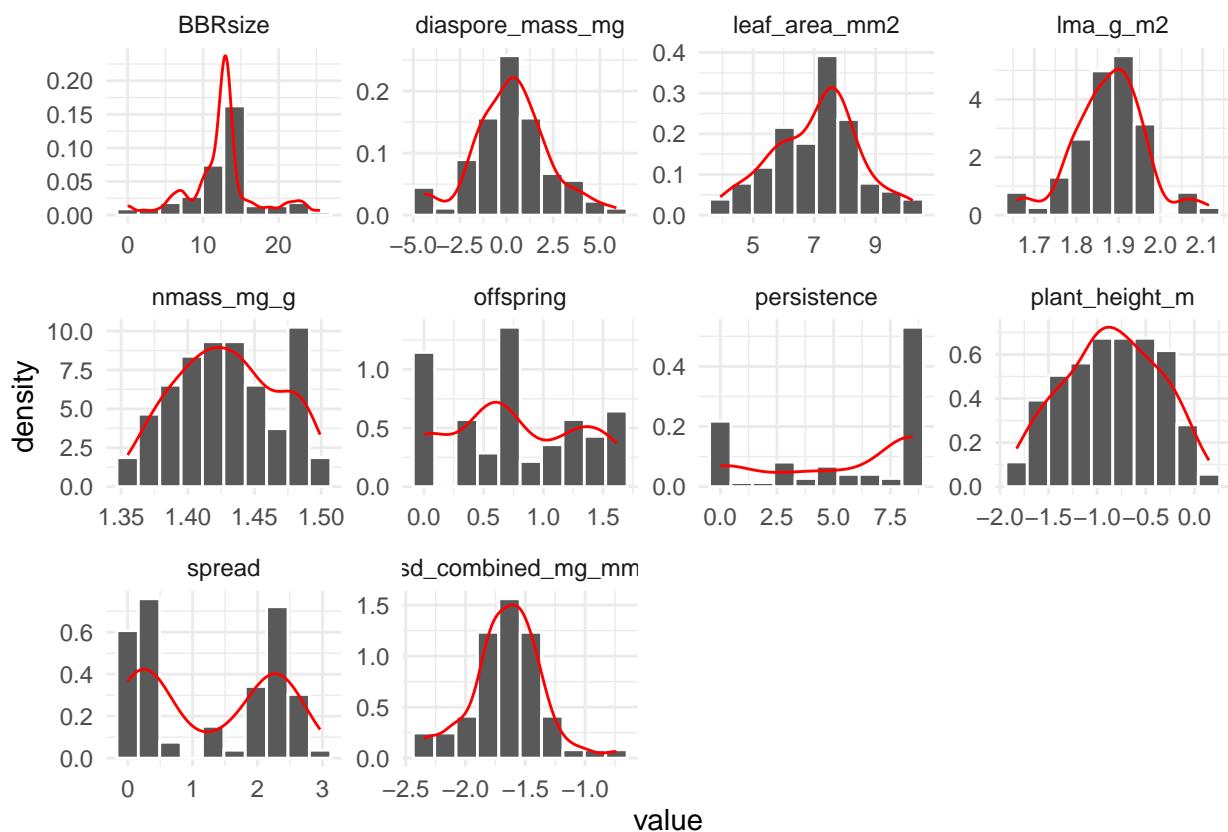
```

```
# check trait distributions

data |>
  pivot_longer(~species, names_to = "trait", values_to = "value") |>
  histdensity(value) +
  facet_wrap(~ trait, scales = "free") +
  theme_minimal()
```

Warning: Removed 76 rows containing non-finite outside the scale range
(`stat_bin()`).

Warning: Removed 76 rows containing non-finite outside the scale range
(`stat_density()`).



```
# overall -----
## calculate distances -----

distances <- data |>
  filter(!species %in% remove_overl) |>
  select(-1) |>
  vegdist(method = "mahalanobis", diag = T, upper = T) |>
  as.matrix()
```

```

#scaling to 0-1

distances <- distances / max(distances)

#double checking the squared euclidean property

ade4::is.euclid(as.dist(sqrt(distances)))

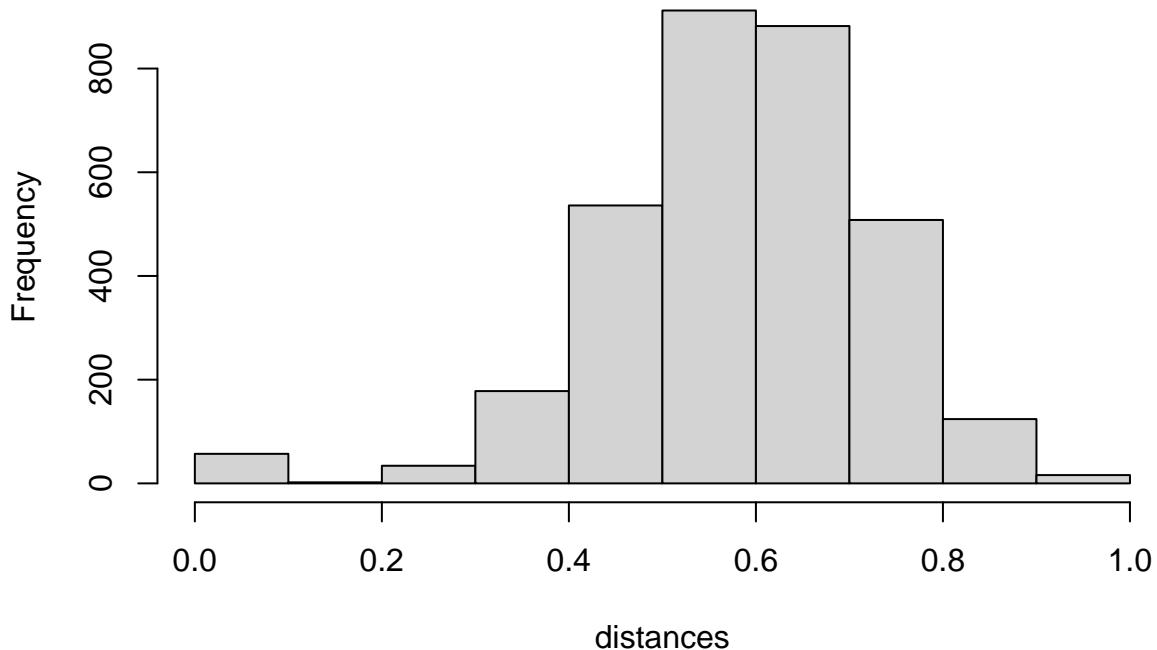
## [1] TRUE

#check distance distribution

hist(distances)

```

Histogram of distances



```

#pairs with distance 0

as.data.frame(as.table(distances)) |>
  filter(Freq == 0 & Var1 != Var2)

## [1] Var1 Var2 Freq
## <0 rows> (or 0-length row.names)

```

```

## calculate indices ----

species <- data |>
  filter(!species %in% remove_overl) |>
  pull(species)

raotrait <- lapply(resampled_overl, function (x)
  lapply(x, function (y) qdecomp(y, distances, species)) |>
    bind_rows(.id = "step") |>
    mutate(step = as.numeric(step))
) |>
  bind_rows(.id = "site")

#scale steps in units of area

raotrait <- raotrait |>
  mutate(step = step / 100)

#add to list
indices[[1]] <- raotrait
names(indices)[1] <- "overall"

## create curve set ----

curvesets <- vector(mode = "list", length = sum(map_lgl(raotrait, is.numeric))-1)

for (i in seq_along(curvesets)){
  column <- colnames(raotrait)[i+2]
  curves <- raotrait |>
    select(site, step, {{column}}) |>
    pivot_wider(names_from = site, values_from = 3) |>
    filter(if_all(where(is.numeric), ~ !is.na(.)))
  curvesets[[i]] <- curve_set(obs = as.data.frame(curves[-1]), r = curves[[1]])
  names(curvesets)[i] <- column
}

indexcurves[[1]] <- curvesets
names(indexcurves)[1] <- "overall"

# aboveground ----

## calculate distances ----

distances <- data |>
  filter(!species %in% remove_above) |>
  select(leaf_area_mm2:ssd_combined_mg_mm3) |>
  vegdist(method = "mahalanobis", diag = T, upper = T) |>
  as.matrix()

#scaling to 0-1

```

```

distances <- distances / max(distances)

#double checking the squared euclidean property

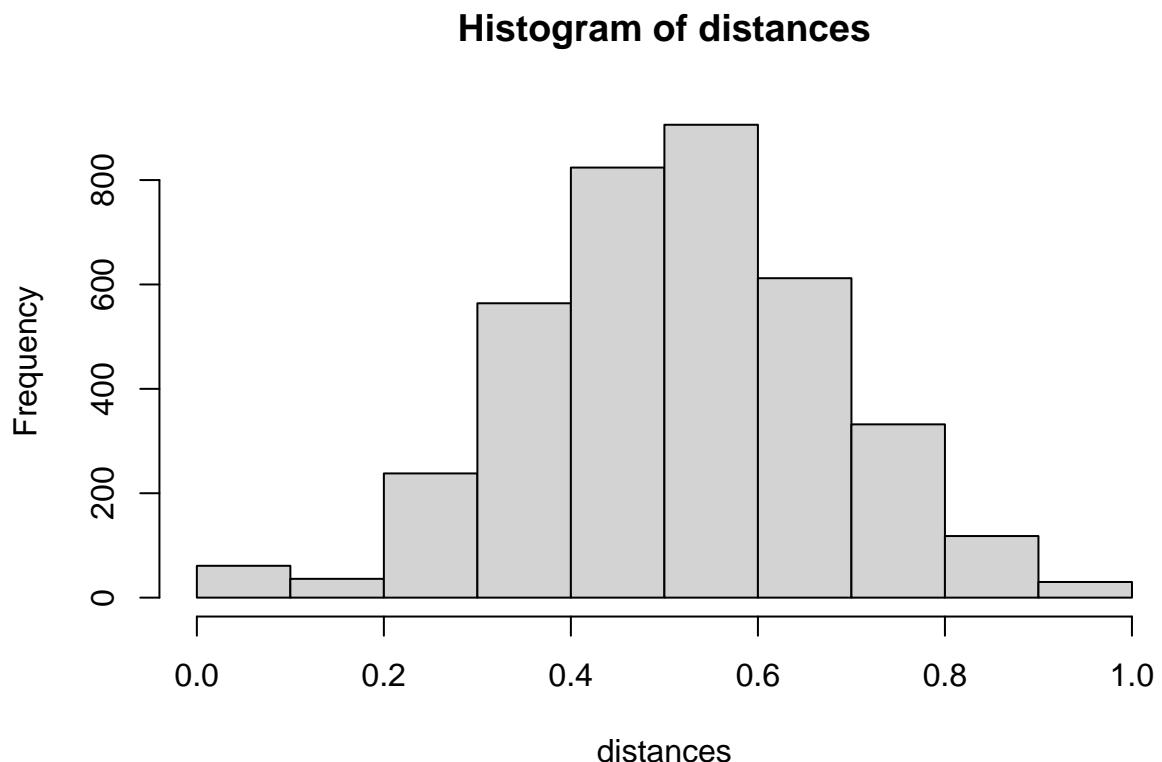
ade4::is.euclid(as.dist(sqrt(distances)))

## [1] TRUE

#check distance distribution

hist(distances)

```



```

#pairs with distance 0

as.data.frame(as.table(distances)) |>
  filter(Freq == 0 & Var1 != Var2)

## [1] Var1 Var2 Freq
## <0 rows> (or 0-length row.names)

## calculate indices -----
species <- data |>

```

```

filter(!species %in% remove_above) |>
  pull(species)

raotrait <- lapply(resampled_above, function (x)
  lapply(x, function (y) qdecomp(y, distances, species)) |>
    bind_rows(.id = "step") |>
    mutate(step = as.numeric(step))
) |>
  bind_rows(.id = "site")

#scale steps in units of area

raotrait <- raotrait |>
  mutate(step = step / 100)

#add to list

indices[[2]] <- raotrait
names(indices)[2] <- "aboveground"

## create curve set -----
curvesets <- vector(mode = "list", length = sum(map_lgl(raotrait, is.numeric))-1)

for (i in seq_along(curvesets)){
  column <- colnames(raotrait)[i+2]
  curves <- raotrait |>
    select(site, step, {{column}}) |>
    pivot_wider(names_from = site, values_from = 3) |>
    filter(if_all(where(is.numeric), ~ !is.na(.)))
  curvesets[[i]] <- curve_set(obs = as.data.frame(curves[-1]), r = curves[[1]])
  names(curvesets)[i] <- column
}

indexcurves[[2]] <- curvesets
names(indexcurves)[2] <- "aboveground"

# clonal -----
## calculate distances -----
distances <- data |>
  filter(!species %in% remove_clo) |>
  select(BBRsize:spread) |>
  vegdist(method = "mahalanobis", diag = T, upper = T) |>
  as.matrix()

#scaling to 0-1

distances <- distances / max(distances)

#double checking the squared euclidean property

```

```

ade4::is.euclid(as.dist(sqrt(distances)))

## Warning in ade4::is.euclid(as.dist(sqrt(distances))): Zero distance(s)

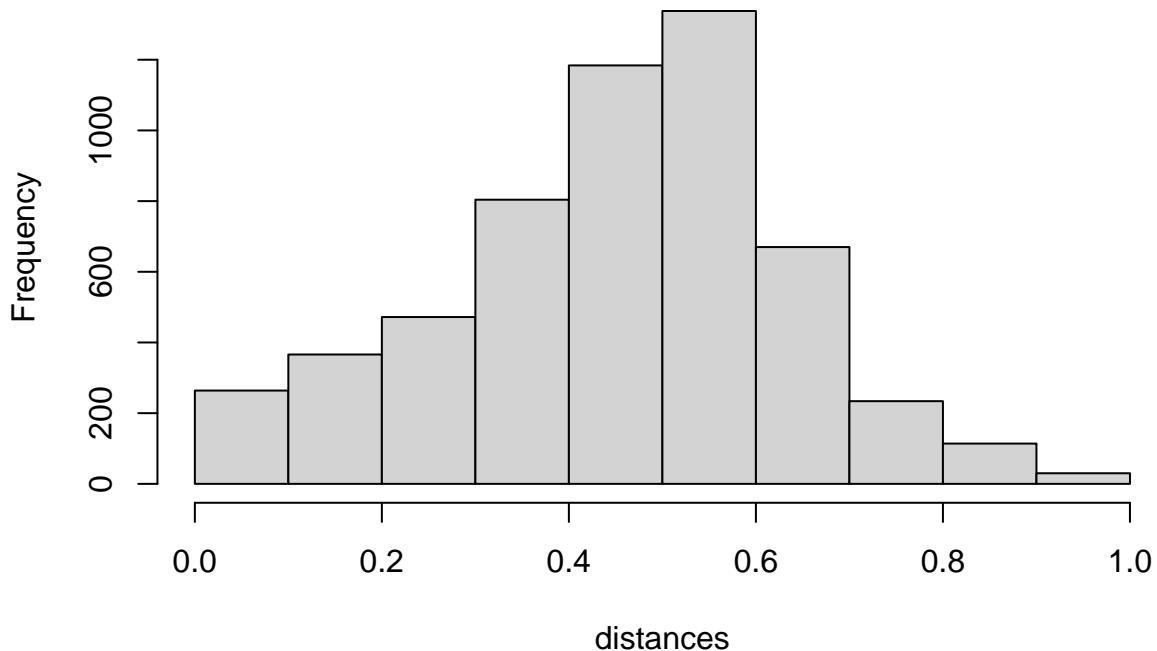
## [1] TRUE

#check distance distribution

hist(distances)

```

Histogram of distances



```

#pairs with distance 0

as.data.frame(as.table(distances)) |>
  filter(Freq == 0 & Var1 != Var2)

```

	Var1	Var2	Freq
## 1	9	1	0
## 2	24	1	0
## 3	38	1	0
## 4	43	1	0
## 5	60	1	0
## 6	1	9	0
## 7	24	9	0
## 8	38	9	0

```
## 9    43    9    0
## 10   60    9    0
## 11   16   13    0
## 12   32   13    0
## 13   29   15    0
## 14   66   15    0
## 15   13   16    0
## 16   32   16    0
## 17   42   23    0
## 18   59   23    0
## 19    1   24    0
## 20    9   24    0
## 21   38   24    0
## 22   43   24    0
## 23   60   24    0
## 24   30   26    0
## 25   31   26    0
## 26   35   26    0
## 27   68   26    0
## 28   15   29    0
## 29   66   29    0
## 30   26   30    0
## 31   31   30    0
## 32   35   30    0
## 33   68   30    0
## 34   26   31    0
## 35   30   31    0
## 36   35   31    0
## 37   68   31    0
## 38   13   32    0
## 39   16   32    0
## 40   55   34    0
## 41   26   35    0
## 42   30   35    0
## 43   31   35    0
## 44   68   35    0
## 45    1   38    0
## 46    9   38    0
## 47   24   38    0
## 48   43   38    0
## 49   60   38    0
## 50   23   42    0
## 51   59   42    0
## 52    1   43    0
## 53    9   43    0
## 54   24   43    0
## 55   38   43    0
## 56   60   43    0
## 57   34   55    0
## 58   23   59    0
## 59   42   59    0
## 60    1   60    0
## 61    9   60    0
## 62   24   60    0
```

```

## 63   38   60   0
## 64   43   60   0
## 65   15   66   0
## 66   29   66   0
## 67   26   68   0
## 68   30   68   0
## 69   31   68   0
## 70   35   68   0
## 71   73   71   0
## 72   71   73   0

## calculate indices -----
species <- data |>
  filter(!species %in% remove_clo) |>
  pull(species)

raotrait <- lapply(resampled_clo, function (x)
  lapply(x, function (y) qdecomp(y, distances, species)) |>
    bind_rows(.id = "step") |>
    mutate(step = as.numeric(step))
) |>
  bind_rows(.id = "site")

#scale steps in units of area

raotrait <- raotrait |>
  mutate(step = step / 100)

#add to list

indices[[3]] <- raotrait
names(indices)[3] <- "clonal"

## create curve set -----
curvesets <- vector(mode = "list", length = sum(map_lgl(raotrait, is.numeric))-1)

for (i in seq_along(curvesets)){
  column <- colnames(raotrait)[i+2]
  curves <- raotrait |>
    select(site, step, {{column}}) |>
    pivot_wider(names_from = site, values_from = 3) |>
    filter(if_all(where(is.numeric), ~ !is.na(.)))
  curvesets[[i]] <- curve_set(obs = as.data.frame(curves[-1]), r = curves[[1]])
  names(curvesets)[i] <- column
}

indexcurves[[3]] <- curvesets
names(indexcurves)[3] <- "clonal"

rm(resampled_above, resampled_clo, resampled_overl)

```

```

# models ----

RNGkind("L'Ecuyer-CMRG")
set.seed(24695)

# Initiate cluster
cl <- makeCluster(detectCores())
parallel::clusterSetRNGStream(cl = cl, iseed = 24695)

#load data

vars <- readRDS("selected_variables.rds")

## FLM testing function ----

flmtest <- function (index, formula, nsim = 19999, cl=NULL){
  graph.flm(nsim = nsim,
    formula.full = formula,
    formula.reduced = Y ~ 1,
    curve_sets = list(Y = index),
    factors = vars,
    cl = cl)
}

## plotting functions ----

#global envelope plot

plot_FLM <- function(x, title = NULL){
  plot <- plot(x) +
    scale_x_continuous( # convert x-axis from area to length
      labels = function(x) x * 10,
      breaks = c(0.01, seq(from = 0.5, to = 2.5, by = 0.5)),
      name = expression("length of sampling units" ~ (italic(m)))
    )
  subtitle <- gsub("Graphical functional GLM: ", "", plot$labels$title)
  plot[["layers"]][[1]][["aes_params"]]$fill <- rgb(188, 223, 235, maxColorValue = 255)
  plot[["layers"]][[1]][["aes_params"]]$alpha <- 1
  plot +
    theme_minimal() +
    labs(title = title,
        subtitle = subtitle) +
    theme(plot.subtitle = element_text(face = if(as.numeric(gsub("p [:punct:]", "", subtitle))>0.05)
      "plain" else "bold", size = 10),
      legend.position = "none",
      strip.text.x = element_text(size = 10))
}

#plot patchwork

patch <- function(x, ncol=2){
  wrap_plots(x, ncol = ncol, byrow = T, guides = "collect") +
    plot_layout(axis_titles = "collect")
}

```

```

}

## select indices from qdecomp output -----
indices <- c("E_alpha", "E_gamma", "E_beta_mult", "redundancy_a", "U_gamma_star", "clustering")

#nicer labels for plotting
labels <- list(E_alpha = expression(E[alpha]),
                E_gamma = expression(E[gamma]),
                E_beta_mult = expression(E[beta]),
                redundancy_a = expression(R[alpha]^"*"),
                U_gamma_star = expression(U[gamma]^"*"),
                clustering = "Clustering")

indexcurves <- lapply(indexcurves, function (x) x[indices])

## fit models -----
flm_FD <- lapply(indexcurves, function (x)
  lapply(x, function (y) flmtest(y, formula = formula(Y ~ RH050 + LAI), cl = cl)))
)

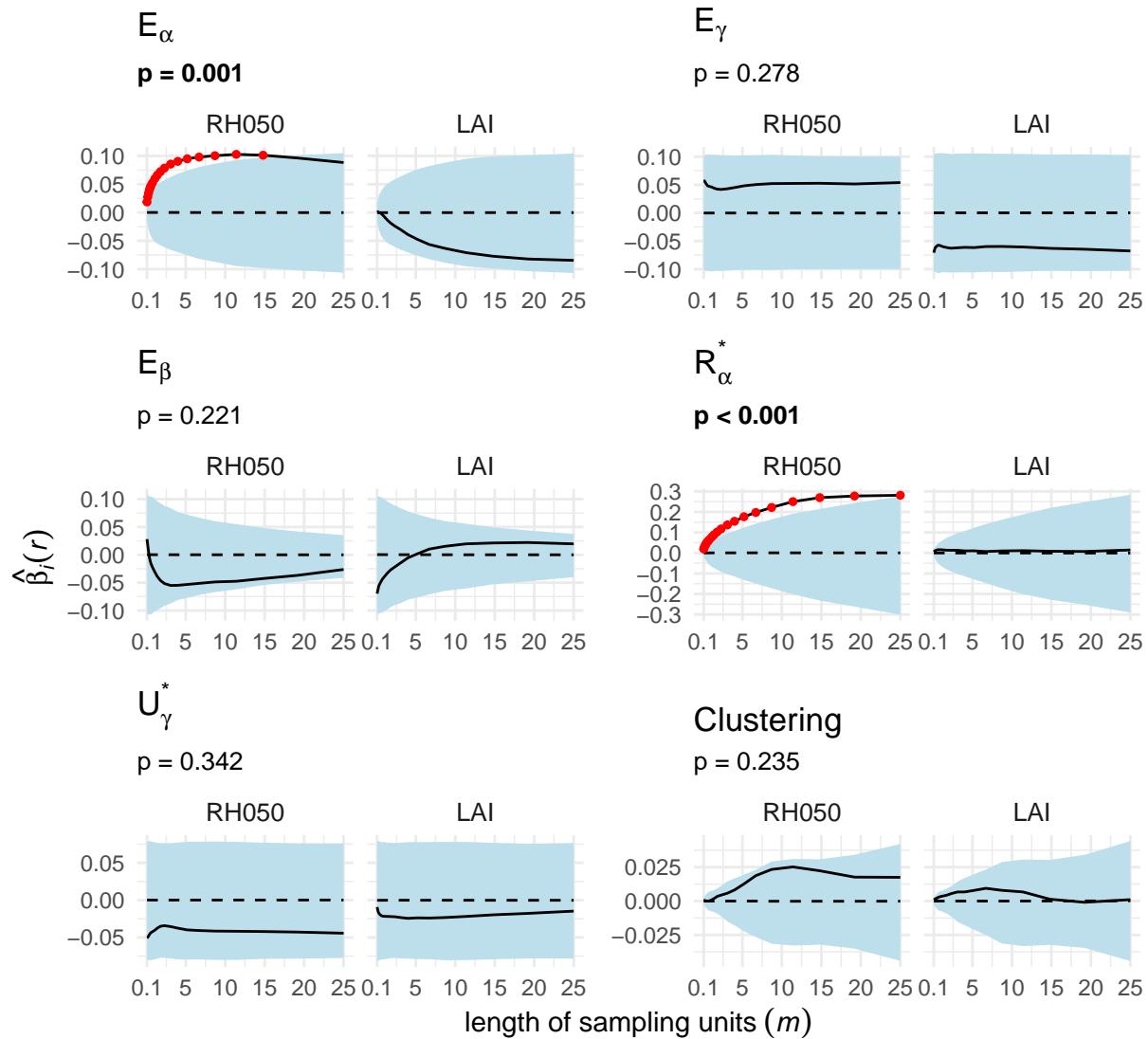
## plotting results -----
plots <- lapply(flm_FD, function (x)
  imap(x, ~ plot_FLM(.x, title = labels[[.y]])))
)

## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## i The deprecated feature was likely used in the GET package.
## Please report the issue at <https://github.com/myllym/GET/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## Warning: `aes_()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i The deprecated feature was likely used in the GET package.
## Please report the issue at <https://github.com/myllym/GET/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

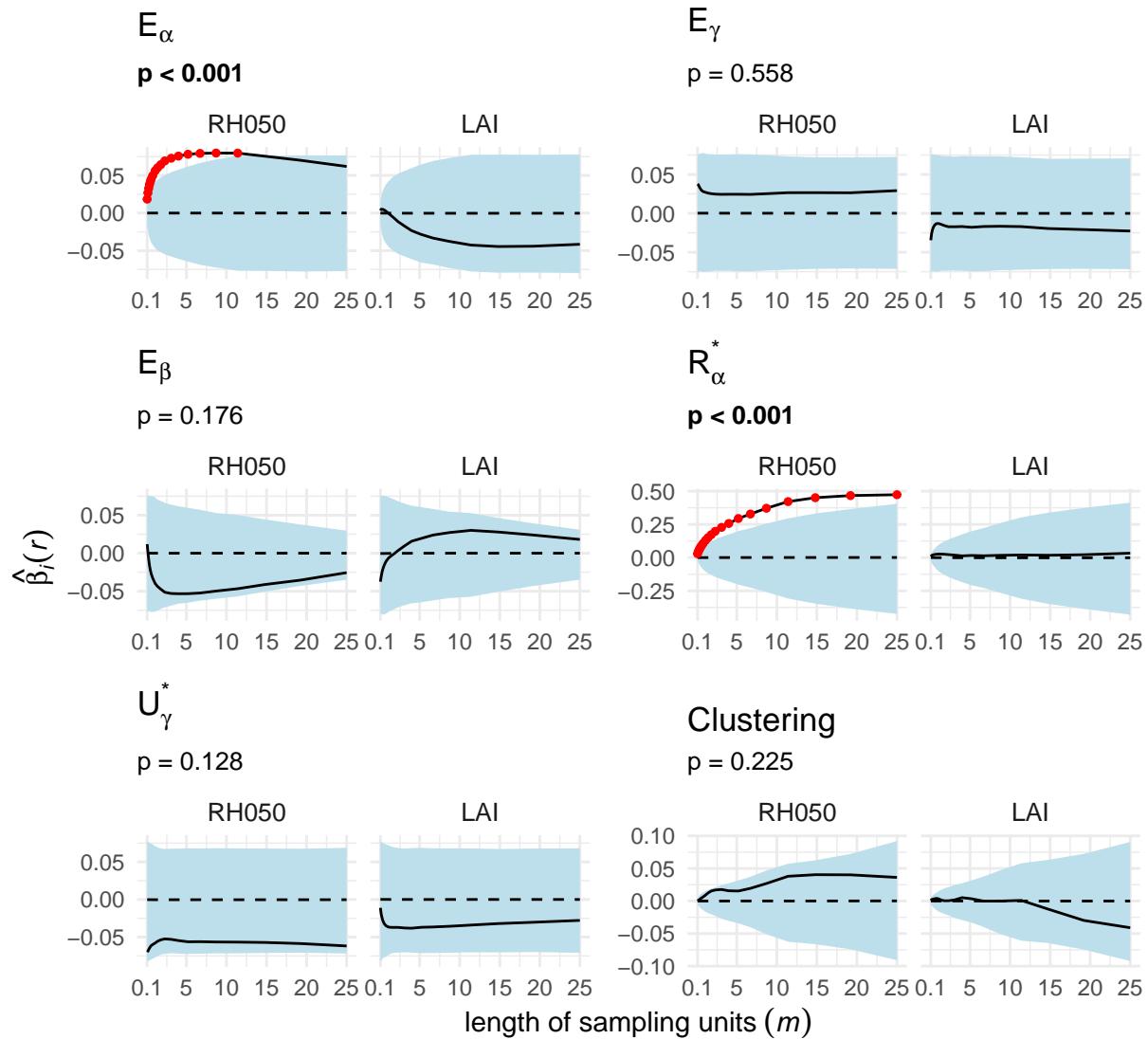
patch(plots$overall)

```



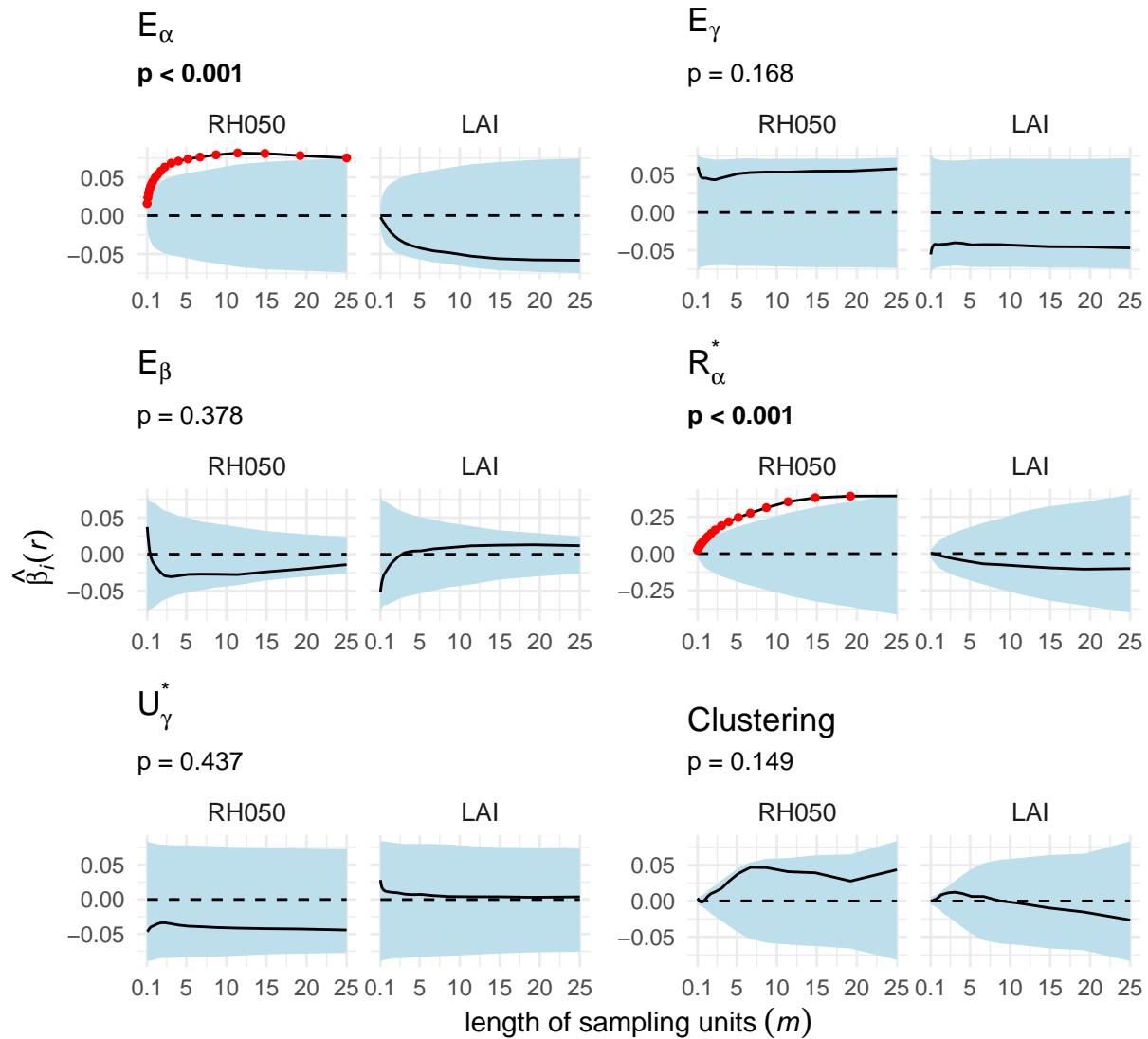
```
ggsave('plots/overall_noimp.png', width = 190, height = 117, units = "mm",
       bg = 'white', scale = 1.5, dpi = 1000)
```

```
patch(plots$aboveground)
```



```
ggsave('plots/above_noimp.png', width = 190, height = 117, units = "mm",
       bg = 'white', scale = 1.5, dpi = 1000)

patch(plots$clonal)
```



```
ggsave('plots/clonal_noimp.png', width = 190, height = 117, units = "mm",
bg = 'white', scale = 1.5, dpi = 1000)
```

```
# session info -----
```

```
sessionInfo()
```

```
## R version 4.5.2 (2025-10-31)
## Platform: x86_64-redhat-linux-gnu
## Running under: Nohara Linux 43 (KDE Plasma Desktop Edition)
##
## Matrix products: default
## BLAS/LAPACK: FlexiBLAS OPENBLAS-OPENMP; LAPACK version 3.12.1
##
## Random number generation:
##   RNG:     L'Ecuyer-CMRG
##   Normal:  Inversion
```

```

## Sample: Rejection
##
## locale:
## [1] LC_CTYPE=it_IT.UTF-8      LC_NUMERIC=C
## [3] LC_TIME=it_IT.utf8        LC_COLLATE=it_IT.UTF-8
## [5] LC_MONETARY=it_IT.utf8    LC_MESSAGES=it_IT.UTF-8
## [7] LC_PAPER=it_IT.utf8       LC_NAME=C
## [9] LC_ADDRESS=C              LC_TELEPHONE=C
## [11] LC_MEASUREMENT=it_IT.utf8 LC_IDENTIFICATION=C
##
## time zone: Europe/Rome
## tzcode source: system (glibc)
##
## attached base packages:
## [1] parallel   stats      graphics  grDevices datasets  utils      methods
## [8] base
##
## other attached packages:
## [1] patchwork_1.3.2 vegan_2.7-2     permute_0.9-8   GET_1.0-7
## [5] lubridate_1.9.4 forcats_1.0.1   stringr_1.6.0   dplyr_1.1.4
## [9] purrrr_1.2.0   readr_2.1.6    tidyverse_2.0.0 tibble_3.3.0
## [13] ggplot2_4.0.1   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
## [1] utf8_1.2.6          generics_0.1.4    stringi_1.8.7    lattice_0.22-7
## [5] hms_1.1.4           digest_0.6.39     magrittr_2.0.4    evaluate_1.0.5
## [9] grid_4.5.2          timechange_0.3.0  RColorBrewer_1.1-3 fastmap_1.2.0
## [13] Matrix_1.7-4       gridExtra_2.3     mgcv_1.9-4       viridisLite_0.4.2
## [17] scales_1.4.0        CoprManager_0.5.7  textshaping_1.0.4 ade4_1.7-23
## [21] cli_3.6.5          crayon_1.5.3     rlang_1.1.6      splines_4.5.2
## [25] withr_3.0.2        yaml_2.3.12      tools_4.5.2     tzdb_0.5.0
## [29] vctrs_0.6.5        R6_2.6.1         lifecycle_1.0.4 MASS_7.3-65
## [33] ragg_1.5.0          cluster_2.1.8.1  pkgconfig_2.0.3  pillar_1.11.1
## [37] gtable_0.3.6        Rcpp_1.1.0        glue_1.8.0       systemfonts_1.3.1
## [41] xfun_0.55           tidyselect_1.2.1  rstudioapi_0.17.1 knitr_1.50
## [45] farver_2.1.2        nlme_3.1-168     htmltools_0.5.9  labeling_0.4.3
## [49] rmarkdown_2.30       compiler_4.5.2    S7_0.2.1

```