Range Weighted Branch Length Difference (RWiBaLD) Calculations & Figure Generation

Nunzio Knerr

2026-02-26

Table of contents

---

1 Introduction

This document contains code for calculating and plotting Range Weighted Branch Length Difference (RWiBaLD). It uses input from Biodiverse analyses of phylogenetic data. The method is related to the Categorical Analysis of Neo And Paleo-Endemisim (CANAPE) published in 2014 and follows up on that paper. Consequently, the acacia dataset from the CANAPE paper is used to illustrate the method.

2 Load packages

Here we load the packages for use in subsequent code.

```
library(sf)
library(ggplot2)
library(Cairo)
library(extrafont)
library(dplyr)
library(ggrepel)
library(stringr)
library(patchwork)
library(forcats)
library(ggtree)
library(ape)
library(knitr)
library(gt)
library(colorBlindness)
```

3 The RWiBaLD metric

The steps for calculating RWiBaLD are as follows:

1. For each branch on the phylogeny (terminal or internal), calculate its *range weighted branch length difference* (RWiBaLD) score as the difference between its length on the RWoT and the RWcT. These scores can be used directly as a continuous measure of neo- and paleo-endemism (negative and positive values, respectively). Classification into neo-endemic, paleo-endemic, and meso-endemic requires further processing using steps 2-4:
2. Identify the "highly endemic branches," i.e., those with range sizes below a threshold. We calculate this threshold by identifying the "elbow" of the distribution, using the *maximum Euclidean distance* method described by Ramer (1972). This is done by ranking all branches according to the inverse of their range size (functionally the same as their lengths on the RWcT), plotting a straight line from the first to last points on that curve, then identifying the point on the curve that produces the longest perpendicular line drawn to it from the straight line (Fig 2A). Branches greater than or equal to the threshold are considered the highly endemic branches of interest.
3. The same elbow statistic is then applied to the RWiBaLD scores from step 1. The positive and negative differences are processed separately (divided at the RWiBaLD = 0 value), resulting in two thresholds (Fig 2B).

4. The highly endemic branches identified in step 2 are then classified using the thresholds from step 3. Highly endemic branches with negative differences less than or equal to the threshold are classified as neo-endemic, those with positive differences greater than or equal to the threshold are classified as paleo-endemic, and those between these two categories are classified as meso-endemic (Fig 2B).

4 Specify biodiverse results data

Specify the tabular data files exported from a biodiverse analysis. Both the observed data & the equal branch length data from a range weighted tree.

```
# Set the data locations etc.
data_dir <- "Acacia_biodiverse_exports/"
observed_data_csv <- paste0(data_dir, "Acacia_RWT_tabular_export.csv")
equal_branch_length_data_csv  <- paste0(data_dir, "Acacia_RWT_EQBL_tabular_export.csv")
```

5 Load data & calculate RWiBaLD

Load the data in, merge the tables and calculate the RWiBaLD score for each branch, then write out the results to a file.

```
# Load data
observed_data <- read.table(observed_data_csv, header=T,sep=",")
equal_branch_length_data <- read.table(equal_branch_length_data_csv, header=T,sep=",")

# Calculate RWiBaLD Statistic etc.
rwibald_results <- observed_data |>
  left_join(equal_branch_length_data, by = 'NAME') |>
  rename(branch_length_observed_tree = LENGTH.x, branch_length_comparison_tree = LENGTH.y) |>
  mutate(rank_comparison_tree = rank(desc(branch_length_comparison_tree), ties.method = "min")) |>
  mutate(sorted_asc_bl_comparison_tree = rank(branch_length_comparison_tree, ties.method = "first")) |>
  mutate(rwibald_score = (branch_length_observed_tree - branch_length_comparison_tree)) |>
  mutate(rwibald_score_rank = rank(rwibald_score, ties.method = "first"))

# View(rwibald_results)
output_dir <- "quarto_outputs/"

# Write data to file
write.csv(rwibald_results, paste0(output_dir, "Acacia_RWiBaLD_results.csv"), row.names=FALSE)
```

6 Elbow point calculation function

We propose a function to calculate the elbow point of a curve in order to determine branches of interest and identify the different categories of RWiBaLD (e.g., neo, meso, paleo).

It takes a numerical vector `data` as input and calculates the point where the rate of decrease in a measure of fit (e.g., sum of squared distances) starts to slow down. The function first orders the values of the data and calculates vectors from the first point to all other points, normalizing the vector from the first to the last point. It then computes the Euclidean distance of each point to this line and returns the y-coordinate (value) of the point with the maximum distance from the line, considered the “elbow” point threshold.

The procedure is described step by step below:

1. **Function Definition and Input:**
   - The function `get_elbow` takes one argument, `data`, which is a numerical vector.
2. **Initial Setup:**
   - `n_pts` is assigned the length of `data`.

- `x_coords` is a sequence from 1 to `n_pts`.
- `y_coords` is the values of `data`, sorted in ascending order.

3. **First Point Coordinates:**
   - `x1` and `y1` are the coordinates of the first point in the sequence.
4. **Normalize the Line Vector:**
   - `x_vec` and `y_vec` are vectors from the first point to all other points.
   - `x_vec_max` and `y_vec_max` are the vectors from the first point to the last point.
   - `normaliser` is the length of the line vector.
   - Normalize `x_vec_max` and `y_vec_max` by dividing by `normaliser`.
5. **Vectors from First Point:**
   - `v_x` and `v_y` are vectors from the first point to all other points.
   - `scalar_prod` calculates the scalar projection of `v_x` and `v_y` onto the normalized line vector.
6. **Calculate Distance to Line:**
   - `vec_to_line_x` and `vec_to_line_y` are the components of the vectors from each point to the line.
   - `dist_to_line` is the Euclidean distance of each point to the line.
   - `i_max` is the index of the maximum distance (the elbow)
7. **Return the Elbow Point Threshold:**
   - The function returns the coordinates (x,y) of the point with the maximum distance from the line, considered the "elbow" point threshold.

In summary, the function identifies the point in the data set that is farthest away from a line drawn between the first and last data points, which often corresponds to a significant change. In case of ties we take the point closest to the minimum.

see: Ramer, U. (1972). *An iterative procedure for the polygonal approximation of plane curves.* **Computer Graphics and Image Processing**, 1, 244–256. https://doi.org/10.1016/S0146-664X(72)80017-0

```
get_elbow <- function(data){

   n_pts = length(data)

   x_coords = 1:n_pts
   y_coords = data[order(data)]

   # First points
   x1 = x_coords[1]
   y1 = y_coords[1]

   # Normalize the line vector
   x_vec = x_coords - x1
   y_vec = y_coords - y1
   x_vec_max = x_vec[n_pts]
   y_vec_max = y_vec[n_pts]

   normaliser = sqrt(x_vec_max^2 + y_vec_max^2)

   x_vec_max = x_vec_max / normaliser
   y_vec_max = y_vec_max / normaliser

   # Vectors from first point
   v_x = x_coords - x1
```

```
    v_y = y_coords - y1

    scalar_prod = v_x * x_vec_max + v_y * y_vec_max

    vec_to_line_x = v_x - scalar_prod * x_vec_max
    vec_to_line_y = v_y - scalar_prod * y_vec_max

    # Distance to line is the norm
    dist_to_line = sqrt(vec_to_line_x^2 + vec_to_line_y^2)

    # y value at the point of maximum distance from the hypotenuse
    #return(y_coords[which(dist_to_line==max(dist_to_line))])

    # Find index of the maximum distance (the elbow)
    i_max <- which.max(dist_to_line)

    # Return both coordinates
    return(list(x = x_coords[i_max], y = y_coords[i_max]))
}
```

7 Calculate branches of interest

Now we take the data and determine the key branches of interest using the elbow point statistic defined in the function above.

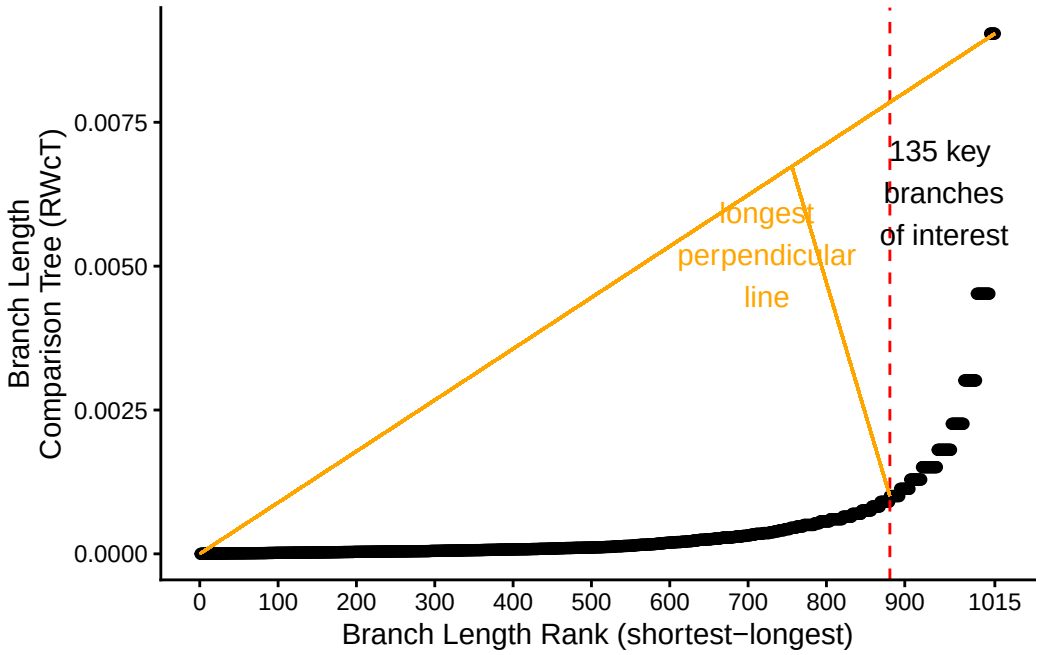
```
# For range weighted tree
df <- rwibald_results[,c("NAME", "rwibald_score", "rwibald_score_rank",
                         "sorted_asc_bl_comparison_tree",
                         "branch_length_comparison_tree",
                         "branch_length_observed_tree")] %>%
      arrange(., branch_length_comparison_tree)

# Call the get_elbow function to find the threshold
elbow_thresh <- get_elbow(df$branch_length_comparison_tree)

# Define key branches
key_branches <- df$branch_length_comparison_tree >= elbow_thresh$y

# Count them
number_of_key_branches <- sum(key_branches)

# Add a column to the results marking the branches of interest as 'sig'
rwibald_results_key <- rwibald_results %>%
  arrange(., branch_length_comparison_tree) %>%
  mutate(bl_comparison_tree_rwibald_elbow_key = ifelse(key_branches, 'key', 'not_key'))

# Merge data
rwibald_results_all <- merge(rwibald_results, rwibald_results_key[,
 ↪  c("NAME","bl_comparison_tree_rwibald_elbow_key")], by=c('NAME','NAME'),all.x=T)

#View(rwibald_results_all)

# Print number of key branches
print(paste0("Number of key branches: ", number_of_key_branches))
```

```
[1] "Number of key branches: 135"
```

8 Generate figure 2A - RWiBaLD branches of interest

Now we plot the ranked branch lengths on the comparison tree showing the cut-off line for the key RWiBaLD branches of interest.

```
# Check the total number of branches
#length(df$branch_length_comparison_tree)

# Add x_marks every 100
x_marks <- c(0, 100, 200, 300, 400, 500, 600, 700, 800, 900, length(df$branch_length_comparison_tree))

branch_length_by_rank <- ggplot(
  df,
  aes(x = as.numeric(rownames(df)), y = sort(branch_length_comparison_tree))
) +
  geom_point() +
  geom_vline(
    xintercept = (length(df$branch_length_comparison_tree) - number_of_key_branches + 1),
    color = "red", linetype = "dashed"
  ) +
  geom_segment(x = 0, y = 0, xend = 1015, yend = 0.0090413023,
    color = "orange", linewidth = 0.5, inherit.aes = FALSE) +
  geom_segment(x = 881, y = 0.001004589, xend = 755.55, yend = 0.00675, color = "orange", linewidth =
   ↪  0.5, inherit.aes = FALSE) +
  annotate("text", x = 724, y = 0.0052, label = "longest\nperpendicular\nline",
           color = "orange") +
  xlab("Branch Length Rank (shortest-longest)") +
  ylab("Branch Length\nComparison Tree (RWcT)") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5)) +
  scale_x_continuous(breaks = x_marks, labels = x_marks) +
  annotate("text", x = 945, y = 0.0072,
           label = paste0(number_of_key_branches, " key\n branches\n of interest"),
           vjust = 1, hjust = 0.5)


print(branch_length_by_rank)
```



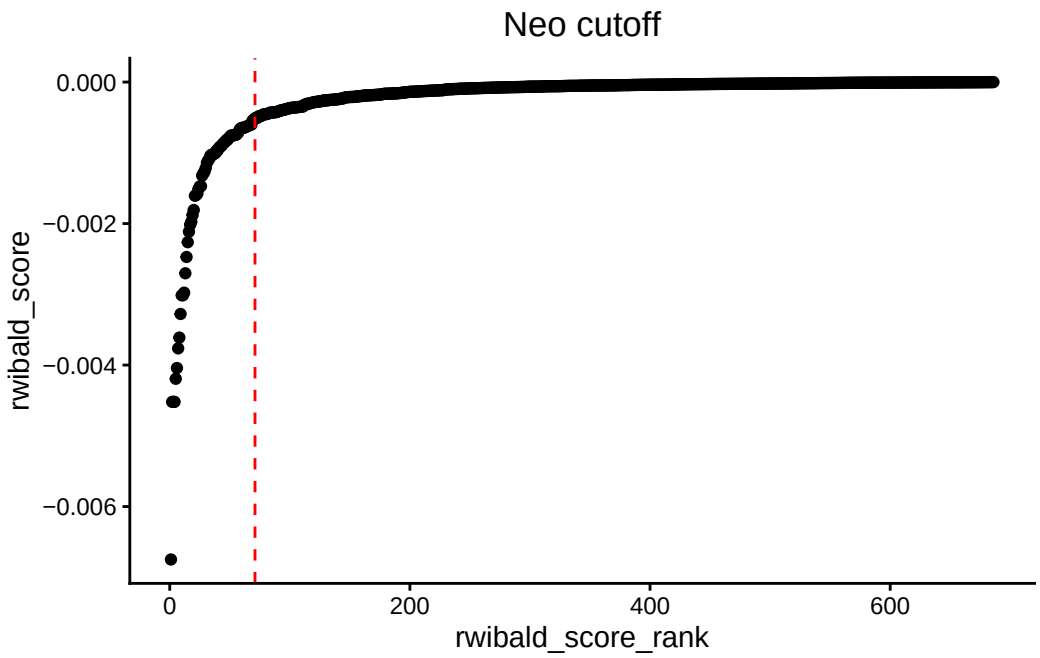

```
output_dir <- "quarto_outputs/"

#cvdPlot(branch_length_by_rank)

ggsave(paste0(output_dir, "figures/Figure2_A.png"), plot = branch_length_by_rank,  scale = 1,
        width = 3000,
        height = 1354,
        units = "px",
        dpi = 300)
```

9 Calculate RWiBaLD branch categories (neo-endemic, meso-endemic, paleo-endemic)

Next we take the key branches of interest and define which of them are neo-endemic, meso-endemic or paleo-endemic. We use the same elbow statistic defined above but only consider the branches of key interest calculated above as valid. We split the data at 0 with those equal or below the elbow point threshold in the negative data being neo-endemics. Those in the positive dataset equal or above the elbow point threshold being classified as paleo-endemic, and rest are classified as meso-endemic. We have described three categories, but of course one could break the continuous distribution into further categories if required.

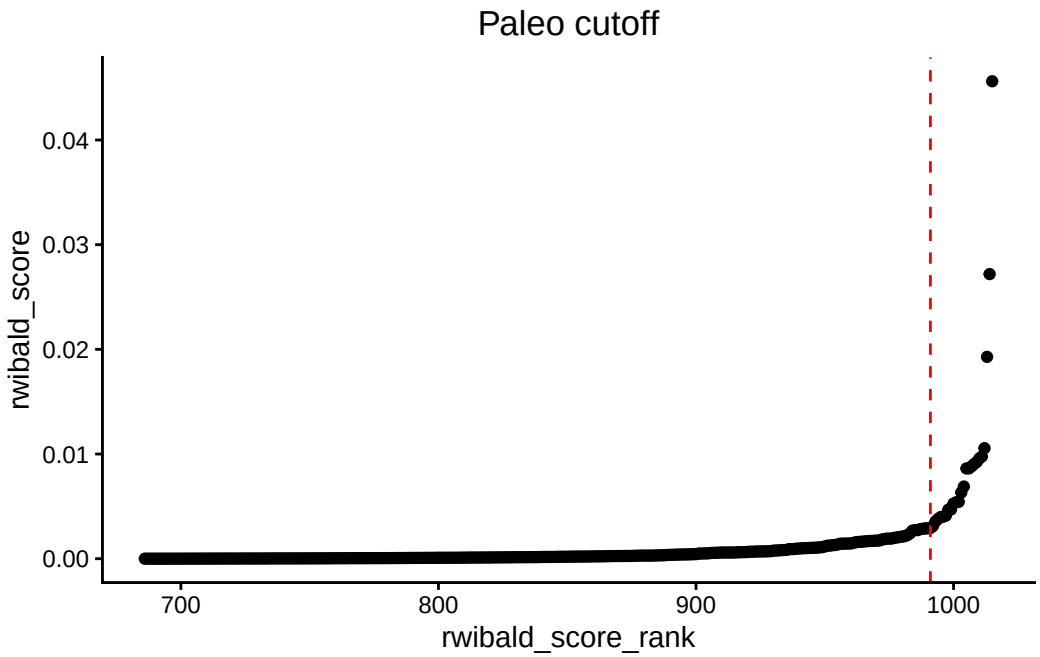
```
# Load the data
data <- rwibald_results_all[,c("NAME", "bl_comparison_tree_rwibald_elbow_key", "rwibald_score",
 ↪  "rwibald_score_rank")] %>%
  arrange(., rwibald_score_rank)

# Subset the left and right dataframes
subset_left_df <- data[data$rwibald_score <= 0, ]
subset_right_df <- data[data$rwibald_score >= 0, ]

# Create a list to store the results for the table
results <- list()

# Records in negative data
negative_records <- nrow(subset_left_df)
results$Negative_Records <- negative_records
```

```r

# Neo cutoff
left_point_x <- get_elbow(subset_left_df$rwibald_score)
rwibald_score_rank_at_left_point <- subset_left_df[subset_left_df$rwibald_score == left_point_x$y,
 ↪  "rwibald_score_rank"]
results$Neo_Cutoff <- rwibald_score_rank_at_left_point

# Plot for left half of curve
l <- ggplot(subset_left_df, aes(x = rwibald_score_rank, y = rwibald_score)) +
  geom_point() +
  geom_vline(xintercept = rwibald_score_rank_at_left_point, color = "red", linetype = "dashed") +
  ggtitle(label= "Neo cutoff") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5))

print(l)
```




```r
ggsave(paste0(output_dir, "figures/Figure2B_L.png"), plot = l,  scale = 1,
       width = 2048,
       height = 1024,
       units = "px",
       dpi = 300)

# Records in positive data
positive_records <- nrow(subset_right_df)
results$Positive_Records <- positive_records

# Paleo cutoff
right_point_x <- get_elbow(subset_right_df$rwibald_score)
rwibald_score_rank_at_right_point <- subset_right_df[subset_right_df$rwibald_score == right_point_x$y,
 ↪  "rwibald_score_rank"]
results$Paleo_Cutoff <- rwibald_score_rank_at_right_point

# Plot for right half of curve
```

```r
r <- ggplot(subset_right_df, aes(x = rwibald_score_rank, y = rwibald_score)) +
  geom_point() +
  geom_vline(xintercept = rwibald_score_rank_at_right_point, color = "red", linetype = "dashed") +
  ggtitle(label= "Paleo cutoff") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5))
print(r)
```




```r
ggsave(paste0(output_dir, "figures/Figure2B_R.png"), plot = r,  scale = 1,
        width = 2048,
        height = 1024,
        units = "px",
        dpi = 300)

# Create a column with the results
rwibald_results_key_type <- rwibald_results_all %>%
  filter(bl_comparison_tree_rwibald_elbow_key == "key") %>%
  mutate(rwibald_type = case_when(rwibald_score_rank <= rwibald_score_rank_at_left_point ~ "neo-endemic",
    ↪
                                  rwibald_score_rank >= rwibald_score_rank_at_right_point ~
 ↪  "paleo-endemic",
                                  TRUE ~ "meso-endemic")) %>%
  select(NAME, rwibald_type)

# ---- Table output: gt for HTML/PDF, kable for GFM ----
summary_results_df <- as.data.frame(t(sapply(results, c)))

if (knitr::is_html_output() || knitr::is_latex_output()) {

  results_gt <- gt::gt(data = summary_results_df) |>
    gt::tab_header(title = "Summary of Records and Cutoffs") |>
    gt::fmt_number(columns = dplyr::everything(), decimals = 0) |>
    gt::tab_style(
      style = gt::cell_text(align = "center"),
```

Summary of Records and Cutoffs

Negative_Records	Neo_Cutoff	Positive_Records	Paleo_Cutoff
686	71	330	991

```
      locations = gt::cells_body()
    )

  results_gt

} else {

  knitr::kable(
    summary_results_df,
    format = "pipe",
    align  = "c"
  )

}

# results_gt <- gt(data = summary_results_df)
#
# # Display the table
# results_gt %>%
#   tab_header(title = "Summary of Records and Cutoffs") %>%
#   fmt_number(columns = everything(), decimals = 0) %>%
#   tab_style(style = cell_text(align = "center"),
#             locations = cells_body()
#             )
# View(rwibald_results_key_type)

# Merge the data
rwibald_results_all <- merge(rwibald_results_all, rwibald_results_key_type[, c("NAME","rwibald_type")],
 ↪  by=c('NAME','NAME'),all.x=T)

# View(rwibald_results_all)
output_dir <- "quarto_outputs/"

# Write data to file
write.csv(rwibald_results_all,paste0(output_dir, "Acacia_RWiBaLD_results_all.csv") ,row.names=FALSE)
```

10 Load RWiBaLD results & tree data by cells

Now we load the data for the RWiBaLD results and the range weighted tree data in tabular format and then replace the counts in the matrix with the branch lengths for that group so we can generate the histograms for each grid cell location. We also add the CANAPE scores to the data table for comparison and calculate the ranges of each branch.

```
data_dir <- "Acacia_biodiverse_exports/"
output_dir <- "quarto_outputs/"

# Data with internal branches (spatial anaylsis from BD) x grid cells
all_tree_data_csv <- paste0(data_dir, "Acacia_PD_Included_Node_List.csv")
RWiBaLD_results_csv  <- paste0(output_dir, "Acacia_RWiBaLD_results_all.csv")

```

```r
all_tree_data <- read.table(all_tree_data_csv, header=T, sep=",", check.names = FALSE )
RWiBaLD_results <- read.table(RWiBaLD_results_csv, header=T, sep=",")

#View(all_tree_data)
#View(RWiBaLD_results)
valueToUse <- "rwibald_score"

# Iterate over the columns in Table 1
for (col in colnames(all_tree_data)[-c(1:3)]) {
  # get the corresponding "valueToUse" column specified above from Table 2
  branch_length <- RWiBaLD_results[RWiBaLD_results$NAME == col, paste0(valueToUse)]
  # replace the numbers in Table 1 with the "valueToUse" value
  all_tree_data[, col] <- ifelse(is.na(all_tree_data[, col]), NA, as.numeric(branch_length))
}

# View(all_tree_data)

# Write data to file
write.csv(all_tree_data, paste0(output_dir, "Acacia_PD_Included_Node_List_", valueToUse, ".csv"),
 ↪  row.names=FALSE)

# Get CANAPE data from biodiverse as well
canape_csv <- paste0(data_dir, "Acacia_Rand1_CANAPE_Export.csv")
canape_results <- read.table(canape_csv, header=T,sep=",")

# View(canape_results)

canape_group_data <- canape_results %>%
                     left_join(all_tree_data %>%
                               select(-c(Axis_0, Axis_1)), by = "ELEMENT"
                               )

# Transpose the dataframe
data <- t(canape_group_data)

# View(data)

# Set the first row as column names
transposed_df <- setNames(data.frame(data[-1,]), data[1,])

# View(transposed_df)

# Merge CANAPE and RWiBaLD data
canape_group_data_rwibald <- transposed_df %>%
  mutate(range_cell_count = rowSums(!is.na(.))) %>%
  tibble::rownames_to_column(var = "NAME") %>%
  left_join(select(rwibald_results_all, NAME, rwibald_type), by = 'NAME') %>%
  mutate(rwibald_type = ifelse(is.na(rwibald_type), "other", rwibald_type))

# Set row names
rownames(canape_group_data_rwibald) <- canape_group_data_rwibald$NAME

# View(canape_group_data_rwibald)

# Add range data to results and create three new columns
rwibald_results_all_with_range <- rwibald_results_all %>%
  left_join(select(canape_group_data_rwibald, NAME, range_cell_count), by = "NAME") %>%
  mutate(rwibald_type = replace(rwibald_type, is.na(rwibald_type), "other"),
         non_range_weighted_branch_length_observed_tree = branch_length_observed_tree * range_cell_count,
```

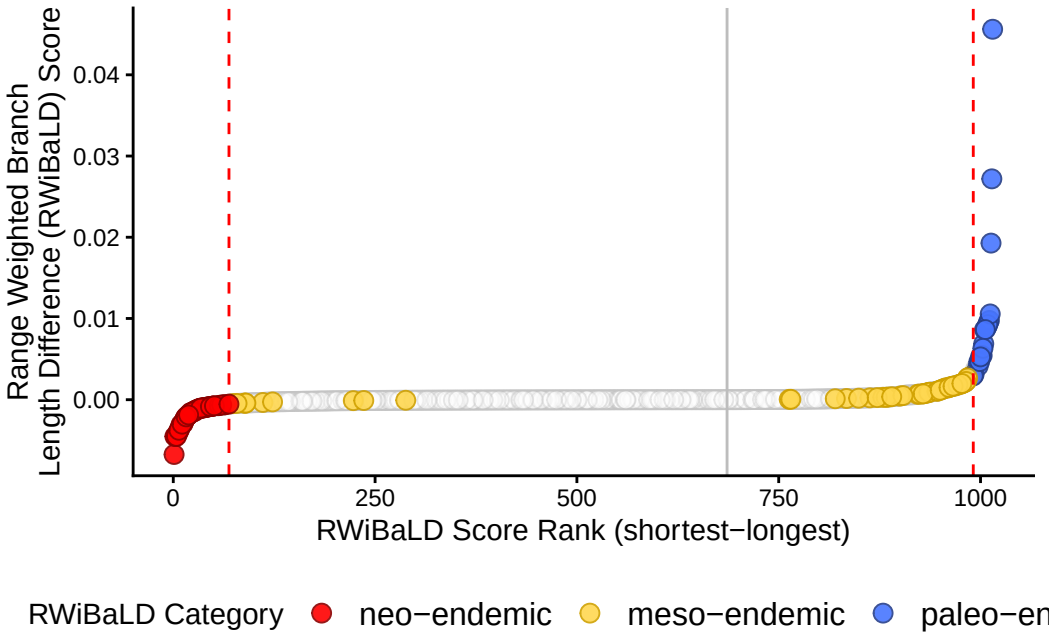
```
         non_range_weighted_branch_length_comparison_tree = branch_length_comparison_tree *
          ↪ range_cell_count,
         diff_non_range_weighted_observed_to_comparison_bl =
          ↪ non_range_weighted_branch_length_observed_tree -
          ↪ non_range_weighted_branch_length_comparison_tree
        ) %>%
  mutate(rank_BaLD = rank(diff_non_range_weighted_observed_to_comparison_bl, ties.method = "first"))


# Write data to file
write.csv(rwibald_results_all_with_range, paste0(output_dir,
 ↪ "Acacia_RWiBaLD_results_all_with_range.csv"), row.names=FALSE)

# View(rwibald_results_all_with_range)
# View(rwibald_results_all)
```

11 Generate figure 2B - ranked RWiBaLD score by RWiBaLD score

Here we plot the ranked RWiBaLD score (x) by the RWiBaLD score (y) (Figure 2B), colouring the different RWiBaLD Categories: neo-endemic (red), meso-endemic (#FFD851) and paleo-endemic (royalblue1).

```
# Find the neo_cutoff
neo_cutoff <- rwibald_results_all_with_range %>%
  filter(rwibald_type == "neo-endemic") %>%
  summarize(highest_rank = max(rwibald_score_rank, na.rm = TRUE)) %>%
  pull(highest_rank)

# Find the paleo_cutoff
paleo_cutoff <- rwibald_results_all_with_range %>%
  filter(rwibald_type == "paleo-endemic") %>%
  summarize(lowest_rank = min(rwibald_score_rank, na.rm = TRUE)) %>%
  pull(lowest_rank)

# Get only the rwibald branches of key interest
rwibald_results_all_key_only <- rwibald_results_all_with_range %>%
                                filter(bl_comparison_tree_rwibald_elbow_key == "key")

# View(rwibald_results_all_key_only)
# View(rwibald_results_all_with_range)

# Plot rwibald rank X rwibald score
rwibald_key_plot <- ggplot() +
    geom_point(data=rwibald_results_all_with_range, aes(x = rwibald_score_rank, y = rwibald_score),
     ↪ fill="transparent", size = 3, colour= "grey77", alpha = 0.4, pch = 21) +
  geom_point(data = rwibald_results_all_key_only, aes(x = rwibald_score_rank, y = rwibald_score, fill =
   ↪ rwibald_type, color = rwibald_type), size = 3, alpha = 0.9, pch = 21) +
    scale_fill_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue1", "neo-endemic"
     ↪ = "red", "meso-endemic" = "#FFD851"), labels = c("neo-endemic","meso-endemic", "paleo-endemic"),
     ↪ limits = c("neo-endemic","meso-endemic", "paleo-endemic"),  na.value = "transparent") +
  scale_color_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue4", "neo-endemic"
   ↪ = "red4", "meso-endemic" = "#D0A100"), labels = c("neo-endemic", "meso-endemic", "paleo-endemic"),
   ↪ limits = c("neo-endemic", "meso-endemic", "paleo-endemic"), na.value = "transparent") +
  geom_vline(xintercept = neo_cutoff, color = "red", linetype = "dashed") +
  geom_vline(xintercept = paleo_cutoff, color = "red", linetype = "dashed") +
  xlab("RWiBaLD Score Rank (shortest-longest)") +
  ylab("Range Weighted Branch\nLength Difference (RWiBaLD) Score") +
 #annotate("segment", x = results$Negative_Records, xend = results$Negative_Records,
```

```
  #          y = max(rwibald_results_all_with_range$rwibald_score) * 1.1, yend = 0,
  #          arrow = arrow(length = unit(0.2, "cm")), color = "grey74") +
  geom_vline(xintercept = results$Negative_Records, color = "grey74", linetype = "solid") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5), legend.position="bottom", legend.text = element_text(size
   ↪  = 12))

print(rwibald_key_plot)
```



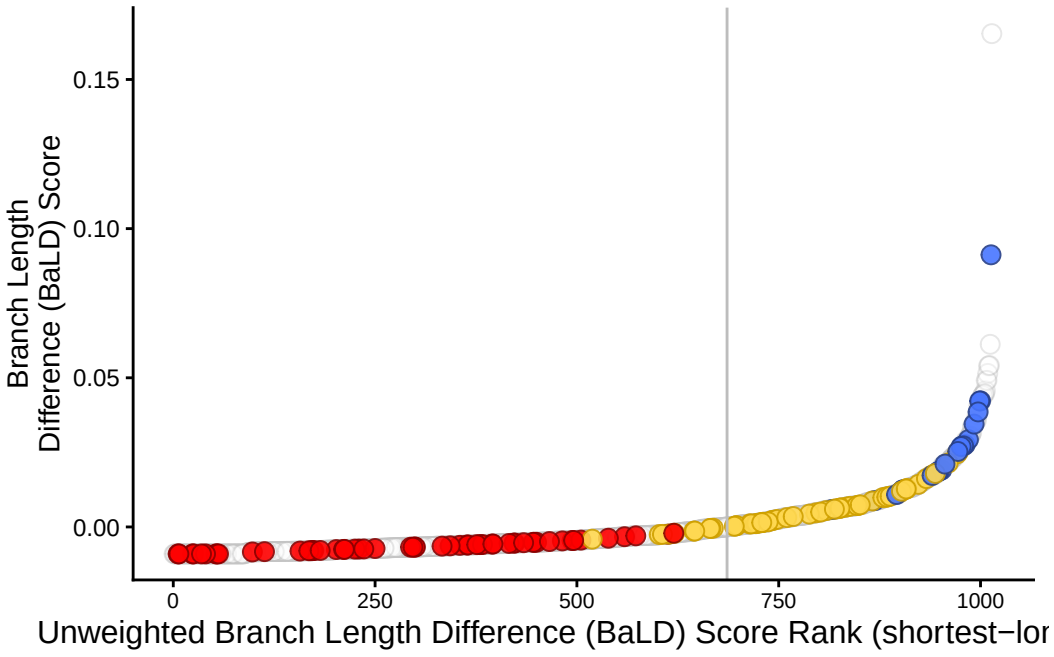

```
#cvdPlot(rwibald_key_plot)

ggsave("quarto_outputs/figures/Figure2_B.png", plot = rwibald_key_plot,  scale = 1,
        width = 3000,
        height = 1354,
        units = "px",
        dpi = 300)
```

12 Generate figure 2C - ranked BaLD by BaLD (non range weighted)

Illustrating the distribution of branch lengths on the observed tree with no range-weighting; the Y-axis is the difference between length on the observed tree and length on the comparison tree for each branch; the X-axis is ranked branch length difference (shortest - longest). The branches of interest are colored by RWiBaLD category as in 2B

```
#View(rwibald_results_all_with_range)

#View(rwibald_results_all_key_only)

# Plot non_range_weighted score
rwibald_key_plot_non_range_weighted <- ggplot() +
    #geom_point(data=rwibald_results_all_with_range, aes(x =
     ↪  reorder(non_range_weighted_branch_length_observed_tree,
     ↪  non_range_weighted_branch_length_observed_tree), y =
     ↪  diff_non_range_weighted_observed_to_comparison_bl), fill="transparent", size = 3, colour=
     ↪  "grey77", alpha = 0.4, pch = 21) +
```

```
  geom_point(data=rwibald_results_all_with_range, aes(x = rank_BaLD, y =
   ↪ diff_non_range_weighted_observed_to_comparison_bl), fill="transparent", size = 3, colour= "grey77",
   ↪ alpha = 0.4, pch = 21) +
  geom_point(data = rwibald_results_all_key_only, aes(x =  rank_BaLD, y =
   ↪ diff_non_range_weighted_observed_to_comparison_bl, fill = rwibald_type, color = rwibald_type), size
   ↪ = 3, alpha = 0.9, pch = 21) +
  scale_fill_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue1", "neo-endemic" =
   ↪ "red", "meso-endemic" = "#FFD851"), labels = c("neo-endemic","meso-endemic", "paleo-endemic"),
   ↪ limits = c("neo-endemic","meso-endemic", "paleo-endemic"),  na.value = "transparent") +
  scale_color_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue4", "neo-endemic"
   ↪ = "red4", "meso-endemic" = "#D0A100"), labels = c("neo-endemic", "meso-endemic", "paleo-endemic"),
   ↪ limits = c("neo-endemic", "meso-endemic", "paleo-endemic"), na.value = "transparent") +
  #geom_vline(xintercept = neo_cutoff, color = "red", linetype = "dashed") +
  #geom_vline(xintercept = paleo_cutoff, color = "red", linetype = "dashed") +
  xlab("Unweighted Branch Length Difference (BaLD) Score Rank (shortest-longest)") +
  ylab("Branch Length\nDifference (BaLD) Score") +
  geom_vline(xintercept = results$Negative_Records, color = "grey74", linetype = "solid") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5), legend.position="none", legend.text = element_text(size =
   ↪ 12), axis.title.x = element_text(size = 12), axis.text.x = element_text(size = 8))

print(rwibald_key_plot_non_range_weighted)
```

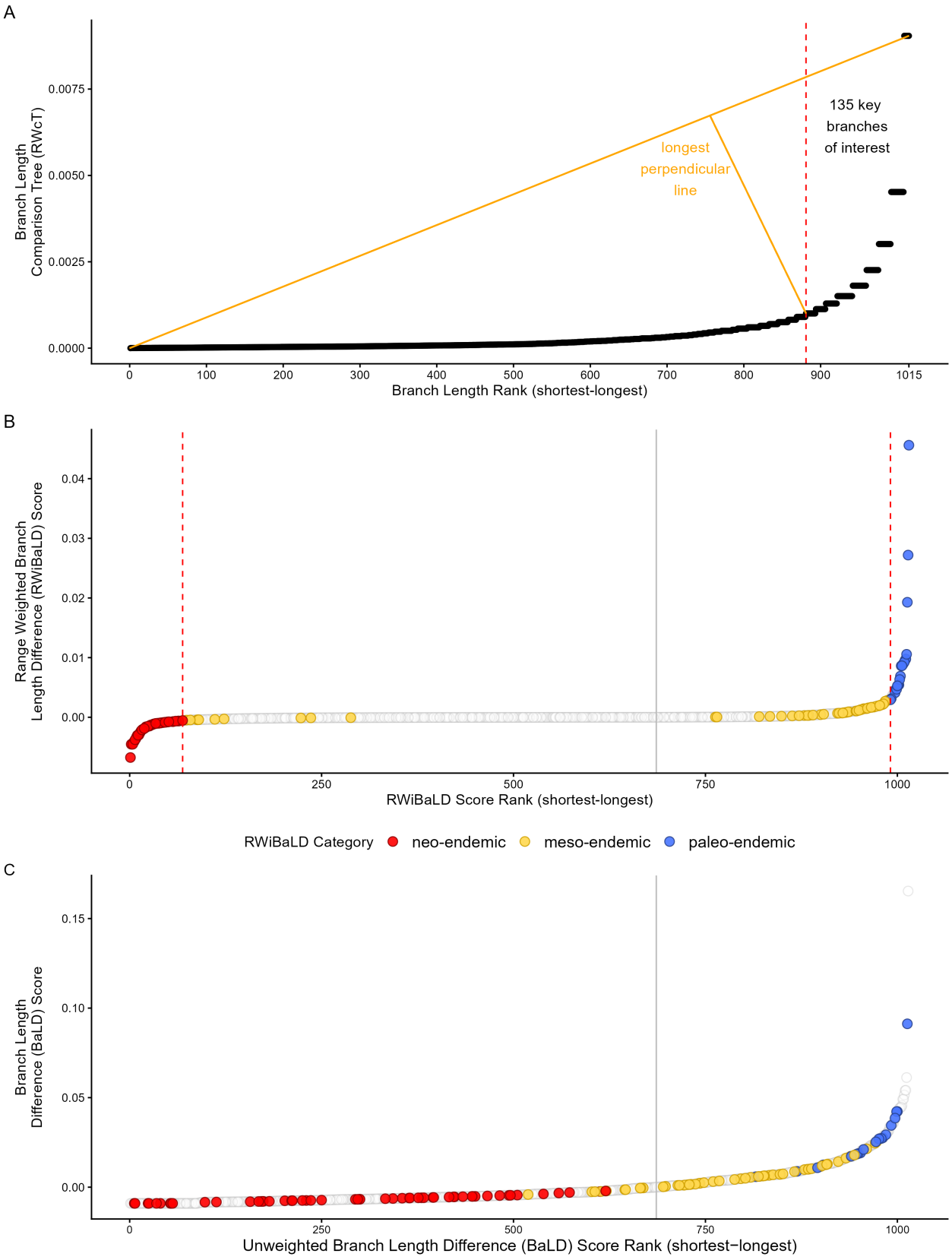



```
ggsave("quarto_outputs/figures/Figure2_C.png", plot = rwibald_key_plot_non_range_weighted,  scale = 1,
       width = 3000,
       height = 1354,
       units = "px",
       dpi = 300)
```

13 Generate multi-panel figure 2ABC

Here we generate a 2 up figure using patchwork

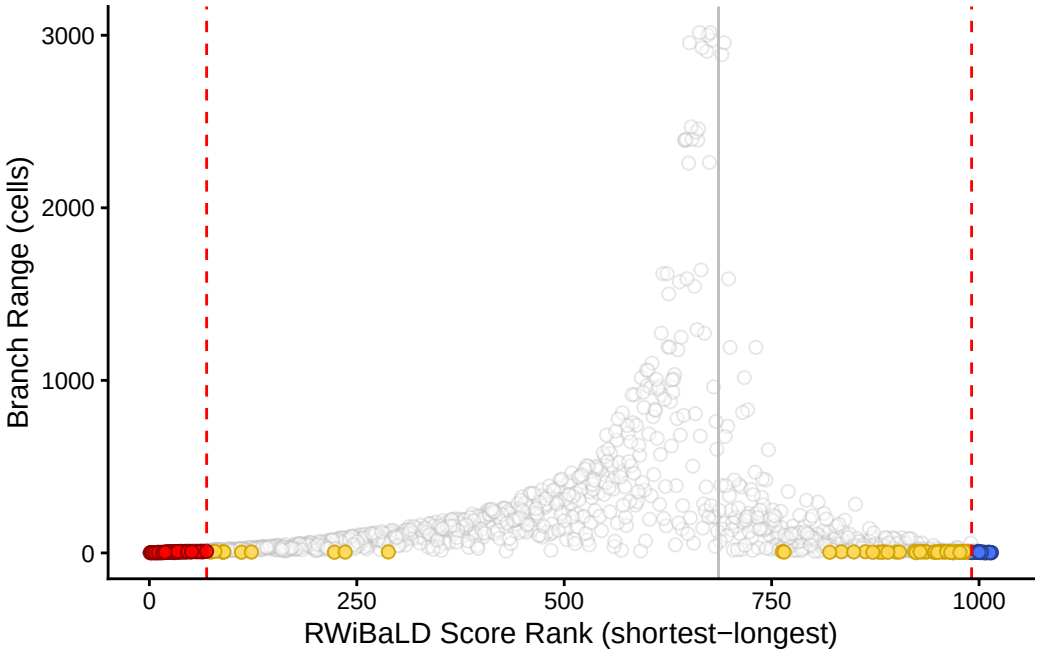
```r
patchwork <- branch_length_by_rank / rwibald_key_plot / rwibald_key_plot_non_range_weighted + plot_annotation(
  #title = 'Branches of Interest for Range Weighted Branch Length Difference (RWiBaLD)\n&\nRWiBaLD Scores With Categories Calculated Using Elbow Statistic',
  theme = theme(plot.title = element_text(hjust = 0.5)),
  subtitle = '',
  caption = '',
tag_levels = 'A') +
  theme(plot.tag.position = c(0, 1),
        plot.tag = element_text(size = 12, hjust = 0, vjust = 0))

#print(patchwork)
#4062
ggsave("quarto_outputs/figures/Figure2_ABC.png", patchwork, width = 3000, height = 4062, units = "px")
```


A
Branch Length
Comparison Tree (RWcT)
0.0075
0.0050
0.0025
0.0000
longest
perpendicular
line
135 key
branches
of interest
0
100
200
300
400
500
600
700
800
900
1015
Branch Length Rank (shortest-longest)
B
Range Weighted Branch
Length Difference (RWiBaLD) Score
0.04
0.03
0.02
0.01
0.00
0
250
500
750
1000
RWiBaLD Score Rank (shortest-longest)
RWiBaLD Category
neo-endemic
meso-endemic
paleo-endemic
C
Branch Length
Difference (BaLD) Score
0.15
0.10
0.05
0.00
0
250
500
750
1000
Unweighted Branch Length Difference (BaLD) Score Rank (shortest−longest)

14 Generate Figure 3A

Here we create a plot of RWiBaLD score (x) by branch range (cells) (Figure 3A), coloured by RWiBaLD Categories: neo-endemic (red), meso-endemic (green) and paleo-endemic (blue).

```
# Plot RWiBaLD rank X branch range
rwibald_key_plot_range <- ggplot() +
    geom_point(data=rwibald_results_all_with_range, aes(x = rwibald_score_rank, y = range_cell_count),
     ↪  fill="transparent", size = 2, colour= "grey77", alpha = 0.4, pch = 21) +
  geom_point(data = rwibald_results_all_key_only, aes(x = rwibald_score_rank, y = range_cell_count, fill
   ↪  = rwibald_type, color = rwibald_type), size = 2, alpha = 0.9, pch = 21) +
  scale_fill_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue1","neo-endemic" =
   ↪  "red", "meso-endemic" = "#FFD851"), labels = c("neo-endemic","meso-endemic", "paleo-endemic"),
   ↪  limits = c("neo-endemic","meso-endemic", "paleo-endemic"),  na.value = "transparent") +
  scale_color_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue4", "neo-endemic"
   ↪  = "red4", "meso-endemic" = "#D0A100"), labels = c("neo-endemic", "meso-endemic", "paleo-endemic"),
   ↪  limits = c("neo-endemic", "meso-endemic", "paleo-endemic"), na.value = "transparent") +
  geom_vline(xintercept = neo_cutoff, color = "red", linetype = "dashed") +
  geom_vline(xintercept = paleo_cutoff, color = "red", linetype = "dashed") +
  geom_vline(xintercept = results$Negative_Records, color = "grey74", linetype = "solid") +
  xlab("RWiBaLD Score Rank (shortest-longest)") +
  ylab("Branch Range (cells)") +
  theme_classic() +
  theme(plot.title = element_text(hjust =1),
        legend.position="none",
        legend.text = element_text(size = rel(1)),
        legend.title = element_text(size = rel(1)),
        axis.title.x = element_text(size = rel(1)),
        axis.title.y = element_text(size = rel(1)),
        axis.text.x = element_text(size = rel(1)),
        axis.text.y = element_text(size = rel(1)))

print(rwibald_key_plot_range)
```



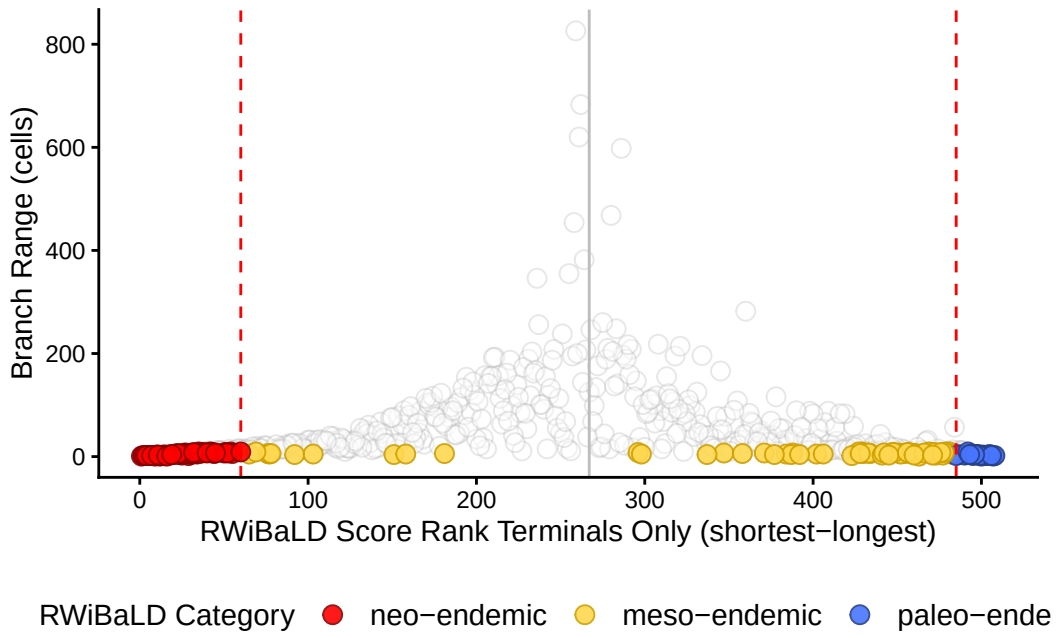
```
ggsave("quarto_outputs/figures/Figure3_A.png", plot = rwibald_key_plot_range,  scale = 1,
        width = 2048,
        height = 1024,
        units = "px",
        dpi = 300)
```

15 Generate Figure 3B

Now the same data as figure 3A but first removing all internal branches before plotting the data. Note: the neo / meso /paleo cuttofs are calculated on the whole dataset and the filtering to terminals done after that.

```
# For plotting purposes get only the rwibald branches of key interest and terminals only
rwibald_results_all_key_only_terminals <- rwibald_results_all_with_range %>%
  filter(!grepl("^[0-9]", NAME)) %>%
  mutate(rwibald_score_rank_terminals = rank(rwibald_score, ties.method = "first")) %>%
    filter(bl_comparison_tree_rwibald_elbow_key == "key")

rwibald_results_all_with_range_terminals <- rwibald_results_all_with_range %>%
    filter(!grepl("^[0-9]", NAME)) %>%
  mutate(rwibald_score_rank_terminals = rank(rwibald_score, ties.method = "first"))

# Find the neo_cutoff
neo_cutoff_terms <- rwibald_results_all_with_range_terminals %>%
  filter(rwibald_type == "neo-endemic") %>%
  summarize(highest_rank = max(rwibald_score_rank_terminals, na.rm = TRUE)) %>%
  pull(highest_rank)

# Find the paleo_cutoff
paleo_cutoff_terms <- rwibald_results_all_with_range_terminals %>%
  filter(rwibald_type == "paleo-endemic") %>%
  summarize(lowest_rank = min(rwibald_score_rank_terminals, na.rm = TRUE)) %>%
  pull(lowest_rank)

# Subset the left and right dataframes
subset_left_df_terms <-
 ↪  rwibald_results_all_with_range_terminals[rwibald_results_all_with_range_terminals$rwibald_score <= 0,
 ↪  ]

# Records in negative data
negative_records_terms <- nrow(subset_left_df_terms)

# View(rwibald_results_all_key_only_terminals)
# View(rwibald_results_all_with_range_terminals)

# View(rwibald_results_all_with_range)

# Plot rwibald rank x rwibald score
# rwibald_key_plot_terminals <- ggplot() +
#     geom_point(data=rwibald_results_all_with_range_terminals, aes(x = rwibald_score_rank_terminals, y =
 ↪  rwibald_score), fill="transparent", size =3, colour= "grey77", alpha = 0.4, pch=21) +
#   geom_point(data = rwibald_results_all_key_only_terminals, aes(x = rwibald_score_rank_terminals, y =
 ↪  rwibald_score, fill = rwibald_type, color = rwibald_type), size =3, alpha = 0.9, pch=21) +
#   scale_fill_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue1", "neo-endemic"
 ↪  = "red", "meso-endemic" = "#FFD851"), labels = c("neo-endemic","meso-endemic", "paleo-endemic"),
 ↪  limits = c("neo-endemic","meso-endemic", "paleo-endemic"),  na.value = "transparent") +
#   scale_color_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue4",
 ↪  "neo-endemic" = "red4", "meso-endemic" = "#D0A100"), labels = c("neo-endemic", "meso-endemic",
 ↪  "paleo-endemic"), limits = c("neo-endemic", "meso-endemic", "paleo-endemic"), na.value =
 ↪  "transparent") +
```

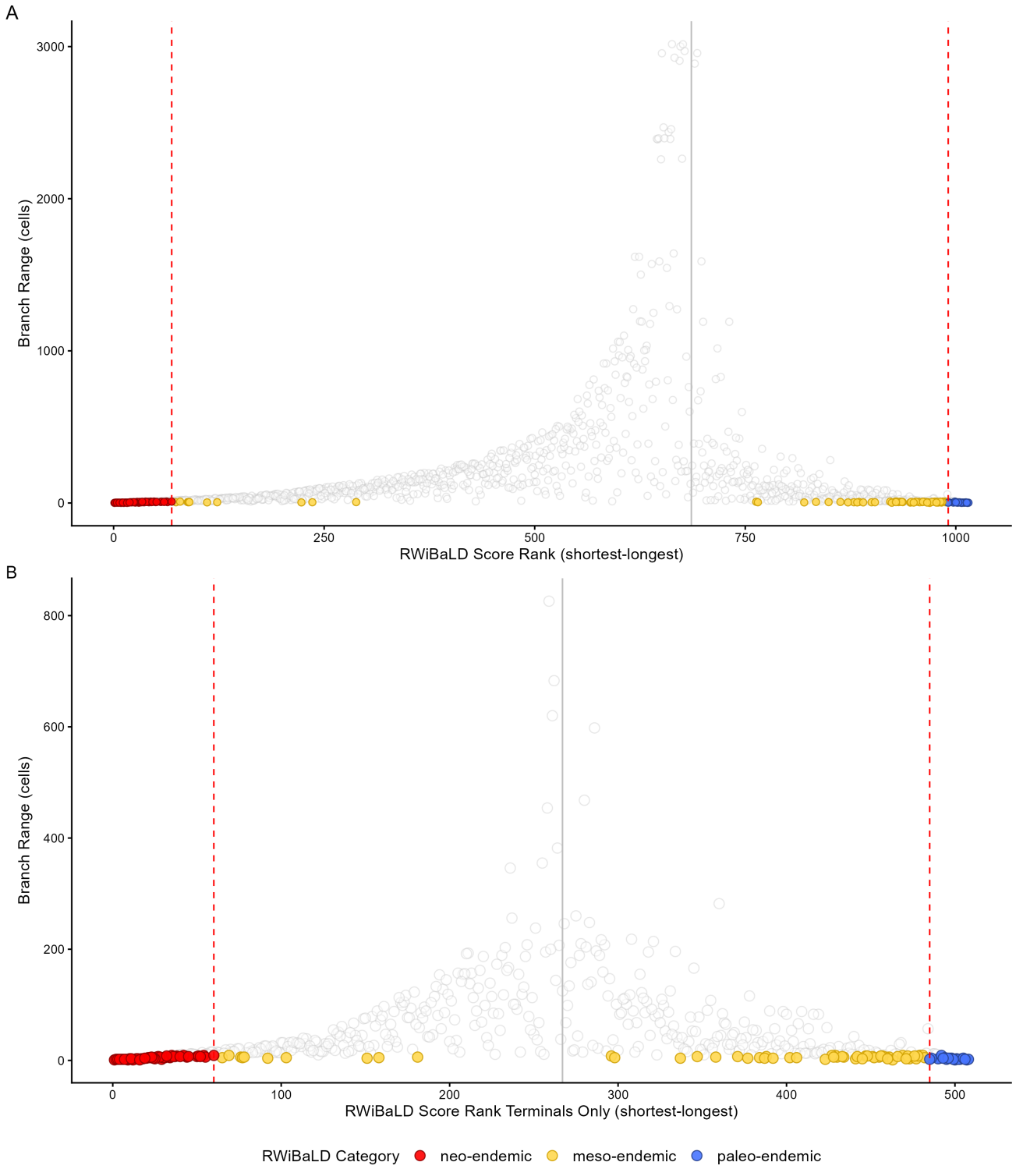
```
#   geom_vline(xintercept = neo_cutoff, color = "red", linetype = "dashed") +
#   geom_vline(xintercept = paleo_cutoff, color = "red", linetype = "dashed") +
#   xlab("RWiBaLD Score Rank Terminals Only (shortest-longest)") +
#   ylab("Range Weighted Branch\nLength Difference (RWiBaLD) Score") +
#   theme_classic() +
#   theme(plot.title = element_text(hjust = 0.5), legend.position="bottom", legend.text =
 ↪  element_text(size = 12))
#
# print(rwibald_key_plot_terminals)
#
# ggsave("quarto_outputs/figures/Figure3_B_terminals_only.png", plot = rwibald_key_plot_terminals,  scale
 ↪  = 1,
#         width = 2048,
#         height = 1024,
#         units = "px",
#         dpi = 300)

# Plot rwibald rank x branch range
rwibald_key_plot_range_terminals <- ggplot() +
    geom_point(data=rwibald_results_all_with_range_terminals, aes(x = rwibald_score_rank_terminals, y =
     ↪  range_cell_count), fill="transparent", size =3, colour= "grey77", alpha = 0.4, pch=21) +
  geom_point(data = rwibald_results_all_key_only_terminals, aes(x = rwibald_score_rank_terminals, y =
   ↪  range_cell_count, fill = rwibald_type, color = rwibald_type), size =3, alpha = 0.9, pch=21) +
  scale_fill_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue1","neo-endemic" =
   ↪  "red", "meso-endemic" = "#FFD851"), labels = c("neo-endemic","meso-endemic", "paleo-endemic"),
   ↪  limits = c("neo-endemic","meso-endemic", "paleo-endemic"),  na.value = "transparent") +
  scale_color_manual(name = "RWiBaLD Category", values = c("paleo-endemic" = "royalblue4", "neo-endemic"
   ↪  = "red4", "meso-endemic" = "#D0A100"), labels = c("neo-endemic", "meso-endemic", "paleo-endemic"),
   ↪  limits = c("neo-endemic", "meso-endemic", "paleo-endemic"), na.value = "transparent") +
  geom_vline(xintercept = neo_cutoff_terms, color = "red", linetype = "dashed") +
  geom_vline(xintercept = paleo_cutoff_terms, color = "red", linetype = "dashed") +
    geom_vline(xintercept = negative_records_terms, color = "grey74", linetype = "solid") +
  xlab("RWiBaLD Score Rank Terminals Only (shortest-longest)") +
  ylab("Branch Range (cells)") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5), legend.position="bottom", legend.text =
   ↪  element_text(size=rel(1)))

print(rwibald_key_plot_range_terminals)
```




```
ggsave("quarto_outputs/figures/Figure3_B.png", plot = rwibald_key_plot_range_terminals,  scale = 1,
        width = 2048,
        height = 1024,
        units = "px",
        dpi = 300)
```

16 Generate multi-panel figure 3AB

```
patchwork <- rwibald_key_plot_range / rwibald_key_plot_range_terminals + plot_annotation(
  theme = theme(plot.title = element_text(hjust = 0.5)),
  subtitle = '',
  caption = '',
tag_levels = 'A') +
   theme(plot.tag.position = c(0, 1),
        plot.tag = element_text(size = 12, hjust = 0, vjust = 0))

#print(patchwork)

ggsave("quarto_outputs/figures/Figure3_AB.png", patchwork, width = 3000, height = 3600, units = "px")
```


A
Branch Range (cells)
3000
2000
1000
0
0
250
500
750
1000
RWiBaLD Score Rank (shortest-longest)
B
Branch Range (cells)
800
600
400
200
0
0
100
200
300
400
500
RWiBaLD Score Rank Terminals Only (shortest-longest)
RWiBaLD Category
neo-endemic
meso-endemic
paleo-endemic

17 Read tree file & generate supplementary figure 1

Now we load the range weighted tree file exported from biodiverse and colour code it by the RWiBaLD categories.

```
# if need be check and install ggtree using this code
# if (!require("BiocManager", quietly = TRUE))
#     install.packages("BiocManager")
#
# BiocManager::install("ggtree")

data_dir <- "Acacia_biodiverse_exports/"

all_tree_data_nwk <- paste0(data_dir, "acacia_tree_TRIMMED1_EQ1_RW1.nwk")

myTree <- read.tree(file=all_tree_data_nwk)

rwibald_results_all <- rwibald_results_all %>%
  mutate(rwibald_type = replace(rwibald_type,is.na(rwibald_type),"other"))

# View(rwibald_results_all)

rwibald_results_all$tree_cols <- as.factor(rwibald_results_all$rwibald_type)

# Remove quotes from labels so they match dataframe
myTree$tip.label <- gsub("'","",myTree$tip.label)
myTree$node.label <- gsub("'","",myTree$node.label)

# Create ggtree object
g <- ggtree(myTree) %<+% rwibald_results_all +
      aes(color=tree_cols) +
      scale_color_manual(values = c("paleo-endemic" = "royalblue1",
                                    "neo-endemic" = "red",
                                    "meso-endemic" = "#FFD851")) +
     # geom_text2(aes(label = node),
     #         hjust =0,    # adjust horizontal position
     #         vjust = 0,     # adjust vertical position
     #         size = 1,        # adjust text size
     #         color = "blue")  + # adjust text color
      theme(legend.position="none") +
       geom_tiplab(as_ylab=FALSE, size=1)

# Display the plot
#print(g)

ggsave("quarto_outputs/figures/SupFigure1_full_tree.png", g, width = 2048,
        height = 4024, units = "px")
```

18 Generate tree in 2 halves supplementary figure 1A and 1B

```
data_dir <- "Acacia_biodiverse_exports/"
all_tree_data_nwk <- paste0(data_dir, "acacia_tree_TRIMMED1_EQ1_RW1.nwk")

myTree <- read.tree(file=all_tree_data_nwk)

rwibald_results_all <- rwibald_results_all %>%
  mutate(rwibald_type = replace(rwibald_type, is.na(rwibald_type), "other"))
```

```r

rwibald_results_all$tree_cols <- as.factor(rwibald_results_all$rwibald_type)

# Remove quotes from labels so they match the dataframe
myTree$tip.label <- gsub("'", "", myTree$tip.label)
myTree$node.label <- gsub("'", "", myTree$node.label)

# Calculate the halfway point
half <- length(myTree$tip.label) / 2

# Sort the tip labels for consistency
tip_labels <- myTree$tip.label
first_half_tips <- tip_labels[1:half]
second_half_tips <- tip_labels[(half + 1):length(tip_labels)]

# Identify MRCA nodes for each half
node_first_half <- getMRCA(myTree, first_half_tips)
node_second_half <- getMRCA(myTree, second_half_tips)


# Create the base ggtree object
g <- ggtree(myTree) %<+% rwibald_results_all +
      aes(color = tree_cols) +
      scale_color_manual(values = c("paleo-endemic" = "royalblue1",
                                    "neo-endemic" = "red",
                                    "meso-endemic" = "#FFD851",
                                    "other" = "gray66")) +
      theme(legend.position = "none") +
      geom_tiplab(as_ylab = FALSE, size=1)

#plot(g)

# Identify MRCA node
mrca_node <- MRCA(g, second_half_tips)

second_half_tree <- viewClade(tree_view = g, mrca_node) +
    geom_point2(
      aes(subset = (node == mrca_node)),
    shape = 21,
    size = 2,
    fill = 'gray26',
    nudge_x = 20,
    position = position_nudge(y = -88))


# +
#   theme(panel.ontop = FALSE,
#         panel.spacing.x = unit(c(1, 1, 1, 1), "cm"),
#         panel.background = element_rect(color = "green", fill="transparent", linewidth = 10),
#         plot.margin = unit(c(1, 1, 1, 1), "cm"))


#plot(second_half_tree)

# Save the figure showing the second half expanded and the first half collapsed
ggsave("quarto_outputs/figures/SupFigure_1A.png",
       second_half_tree, width = 2048, height = 4024, units = "px")

#plot(g)
```

```r
# Collapse the second half of the tree
g_half_collapsed <- collapse(g, node = node_second_half) +
  # Add a point to indicate the collapsed node for reference
  geom_point2(aes(subset=(node == node_second_half)), shape=21, size=2, fill='gray26')

#plot(g_half_collapsed)

# Save the figure showing the first half expanded and the second half collapsed
ggsave("quarto_outputs/figures/SupFigure_1B.png",
       g_half_collapsed, width = 2048, height = 4024, units = "px")
```


hastulata
tybaensis
bemolulata
nuegelii
aenuza
staudtiphyta
costata
desmoides
dusenii
andrewsii
aurocoma
retivenea
duseni
stahlogaea
lateriana
glaucoderma
acoma
hallana
eventricia
bilaria
glaucophlora
thamnophylla
hylaia
pruinocarpa
alaica
cammerum
trachyantha
kaalaia
aaaensis
alaskana
fragilis
brachyphylla
heterocolla
pulcrophylla
complicatiramia
leptoneura
corrugata
denticulata
ketogamophylla
acanthoclada
lechleriana
schrata
arbora
aurantiaca
cavaleriei
acuaria
berthamii
diplophelleia
laburnea
jelskii
meiandra
casimiroa
oreades
distigma
trianaeiana
glauca
psoroi
campylocalyx
anthotriaria
eurenchana
leiophylla
pomifera
triquetra
rekoaides
muntingiana
lasthyana
pseudopyxis
bullegeia
galeana
myrtoloia
crossanthera
stenostephana
hookeriana
pinatoviana
bifora
myrsinoides
pauciflora
jacksonioides
guineti
chaimoceiba
pyxiflora
figuroides
aucuparia
leioderma
glauca
rigidiana
amygdaloides
brownlowia
pentaderma
brunfelsiifolia
alata
duplicata
gonophylla
externa
pavoi
asperuloides
hypodadnia
acuva
arcuata
hyaeniodes
pungens
blakeyi
corymbifera
spathulifolia
xanthina
rostellifera
cuneata
didyma
biverticata
hyaloc
bulleriana
capuronii
bifurcata
salicina
angulosa
pentagyna
aphylla
stenocalyx
woodcockiorum
acanthocladia
crustata
oxyclada
bakeriana
sphenosiaena
confusa
praefolia
pendyrgona
extrorsiflata
serpens
wallichii
sciadiflorens
viscosa
multiflagrum
kirkii
hybridoma
complanata
acuminatissima
cameusiana
longipennis
glaziovii
saligna
baillonii
charlesiana
cauliflora
kyrsonoscia
alexandri
victoriae
congesta
leiocarpophylla
pyrifolia
dampieriana
pauciflora
cuspidifolia
mayanaensis
maculosiana
achariodelpha
adinophylla
semicrenata

19 Load data & generate histograms for all cells

Next we load the data and generate histograms. This code generates a histogram for every cell in the dataset. For convenience the file names contain the CANAPE code and whether the histogram contains branches of key interest.

```
#function to change decimal places displayed on plot
scaleFUN <- function(x) sprintf("%.8g", x)

# Function to replace '-' with 'm' and ':' with '_'
clean_column_names <- function(df) {
  new_names <- colnames(df) %>%
    gsub("-", "neg", .) %>%
    gsub(":", "_", .)
  colnames(df) <- new_names
  return(df)
}

# Clean column names using the function
canape_group_data_rwibald <- clean_column_names(canape_group_data_rwibald)

#View(canape_group_data_rwibald)

col_scheme <- c("paleo-endemic" = "royalblue1","neo-endemic" = "red", "meso-endemic" = "#FFD851", "other"
 ↪  = "lightgoldenrodyellow")
#legend_labels <- c("neo-endemic"="Neo Endemic","paleo-endemic"="paleo Endemic", "other" = "other",
 ↪  "meso-endemic" = "Meso Endemic")

#data <- transposed_df
#View(canape_key_group_data_rwibald)

#Generate all histograms
for (i in 2:(ncol(canape_group_data_rwibald)-2)) {# skip first column and last 2 as they are
 ↪  NAME,cell_count,rwibald_type
#for (i in 74:74) {
  #i <- 74
  data <- canape_group_data_rwibald[-(1:6),] # remove the rows not needed
  col_name <- colnames(data)[i] # get the column name to generate the histogram of
  colourByCol <- "rwibald_type"  # set the column to use for colouring the histogram
  numberOfBins <- 41 # set number of bins to use
  hist_data <- na.omit(as.numeric(data[[col_name]])) # extract that column data for ploting
  hist_range <- range(hist_data,na.rm=TRUE) # calculate the range of all values in the column ie. min/max
 ↪  value

#calculate the total range of the whole data set to get a consistent x scale on all histograms.
dataForRange <- data %>%
  select(-first(colnames(.)), -last(colnames(.))) %>%
  mutate(across(everything(), as.numeric))
totalRange <- range(dataForRange, na.rm=TRUE)#lowest and highedt values in the dataset
totalRangeGap <- max(dataForRange, na.rm = TRUE) - min(dataForRange, na.rm = TRUE) # distance between
 ↪  range above

  range <- max(hist_data) - min(hist_data)
  binSize <- range/numberOfBins
  farthest_number <- max(abs(hist_range)) # calculate the number farthest from 0
  x_min <- -(farthest_number)-2*(binSize) # set xmin so zero is centred
  x_max <- (farthest_number)+2*(binSize) # set xmax so zero is centred

  #check to see if any of the branches in the histogram are RWIBALD significant to add to the file names
  result <- data %>%
```

```
  select(!!sym(col_name), !!sym(colourByCol)) %>%
  filter(!is.na(!!sym(col_name))) %>%
  summarise(result = ifelse(any(!!sym(colourByCol) %in% c("paleo-endemic", "neo-endemic",
   ↪  "meso-endemic")), "key", "zero-key")) %>%
  pull(result)

  #View(result)
  # setup the file names
  CANAPE_CODE <- canape_group_data_rwibald["CANAPE_CODE", col_name]
  cell <- str_replace_all(col_name, "[:]", "_")
  filename <- paste0("quarto_outputs/histograms/", cell, "_CC_", CANAPE_CODE,"_", result, "_",
↪  numberOfBins,"bins.png")

  #print(filename)
  # get a subset of dataframe for the histogram
  plot_data <- data %>%
  select(!!sym(col_name), !!sym(colourByCol)) %>%
  filter(!is.na(!!sym(col_name))) %>%
  arrange(!!sym(colourByCol))

  #View(plot_data)

 plot <- ggplot(plot_data, aes(x = as.numeric(.data[[col_name]]), fill = forcats::fct_rev(rwibald_type)))
↪  +
  geom_histogram(color = "black", linewidth= 0.2, bins = numberOfBins) +
  scale_fill_manual(values = col_scheme) +
  xlim(x_min, x_max) +
  coord_cartesian(ylim = c(0, 15)) +
  labs(x = "RWiBaLD Score", y = "Frequency", fill = "Branch Category") +
  theme_bw()

 plot <- plot +
  stat_bin(
    aes(label = after_stat(if_else (condition = count>15, as.character(count), ""))),
    bins= as.numeric(numberOfBins),
    position=position_stack(vjust=0.1),
    pad=TRUE,
    geom = "text",
    color = "black",
    size = 2,
    y = 14
  )

 CairoPNG(width = 1024, height = 600, file = filename, canvas="white", bg = "white", units="px", dpi=96,
  ↪  title = "")
  print(plot)
 dev.off()
}
```

20 Load data & generate 12 histograms for figure 4A-L

This code filters out only the 12 histograms included in Figure 4 and generates the ggplot2 objects of them.

```
data <- t(canape_group_data)
#View(data)

#Transpose the dataframe and set the first row as column names
```

```r
transposed_df <- setNames(data.frame(data[-1,]), data[1,])

#View(transposed_df)

#cells to keep
cols_to_keep <- c("NAME", "-1825000:-2975000", "75000:-2525000",
 ↪  "625000:-3475000","-1425000:-3225000","1775000:-3725000", "-1425000:-3475000", "-1225000:-3775000",
 ↪  "-1425000:-3275000", "1225000:-1275000", "-1175000:-3275000", "-1075000:-2825000",
 ↪  "-1075000:-2375000")

#Calculate RWiBaLD significants and Statistics etc.
canape_group_data_rwibald <- transposed_df %>%
  tibble::rownames_to_column(var = "NAME") %>%
  select(all_of(cols_to_keep)) %>%
  left_join(select(rwibald_results_all, NAME, rwibald_type), by = 'NAME') %>%
  mutate(rwibald_type = ifelse(is.na(rwibald_type), "other", rwibald_type))

# Check if the column is already a factor
 if (!is.factor(canape_group_data_rwibald$rwibald_type)) {
   # If not a factor, convert and specify levels
   canape_group_data_rwibald$rwibald_type <- factor(canape_group_data_rwibald$rwibald_type, levels =
 ↪  c("neo-endemic", "meso-endemic", "paleo-endemic", "other"))
 }

#View(canape_group_data_rwibald)

# Function to replace '-' with 'm' and ':' with '_'
clean_column_names <- function(df) {
  new_names <- colnames(df) %>%
    gsub("-", "neg", .) %>%
    gsub(":", "_", .)
  colnames(df) <- new_names
  return(df)
}

# Clean column names using the function
canape_group_data_rwibald <- clean_column_names(canape_group_data_rwibald)

rownames(canape_group_data_rwibald) <- canape_group_data_rwibald$NAME

#View(canape_group_data_rwibald)

# Write data to file
write.csv(canape_group_data_rwibald, paste0("quarto_outputs/transposed_df_figure_only.csv"),
 ↪  row.names=FALSE)

#function to change decimal places displayed on plot
scaleFUN <- function(x) sprintf("%.8g", x)

#View(canape_group_data_rwibald)

col_scheme <- c("paleo-endemic" = "royalblue1",
                "neo-endemic" = "red",
                "meso-endemic" = "#FFD851",
                "other" = "lightgoldenrodyellow"
                )

legend_labels <- c("neo-endemic" = "neo-endemic",
                   "paleo-endemic" = "paleo-endemic",
```

```
                    "meso-endemic" = "meso-endemic",
                    "other" = "other"
                    )

legend_order <- c("neo-endemic", "meso-endemic", "paleo-endemic", "other")

#data <- transposed_df
#View(canape_group_data_rwibald)

# Create an empty list to store plots
plot_list <- list()

#Generate histograms
for (i in 2:(ncol(canape_group_data_rwibald)-1)) {# skip first column and last 2 as they are NAME,rwibald_type
  #i <- 2
  local({ # have to make this local to allow multiple plots in patchwork
    i <- i
  data <- canape_group_data_rwibald[-(1:6),] # remove the rows not needed
  col_name <- colnames(data)[i] # get the column name to generate the histogram of
  colourByCol <- "rwibald_type"  # set the column to use for colouring the histogram
  numberOfBins <- 41 # set number of bins to use

  # have to show only one legend as the 'collect' feature in patchwork does not collect the legends as desired
  if(i == 3){
  showLegend <- TRUE
  } else {
  showLegend <- FALSE
  }

  hist_data <- na.omit(as.numeric(data[[col_name]])) # extract that column data for ploting
  hist_range <- range(hist_data,na.rm=TRUE) # calculate the range of all values in the column ie. min/max value

  #calculate the total range of the whole data set to get a consistent x scale on all histograms.
  dataForRange <- data %>%
  select(-first(colnames(.)), -last(colnames(.))) %>%
  mutate(across(everything(), as.numeric))
  totalRange <- range(dataForRange, na.rm=TRUE)#lowest and highest values in the dataset
  totalRangeGap <- max(dataForRange, na.rm = TRUE) - min(dataForRange, na.rm = TRUE) # distance between range above

  range <- max(hist_data) - min(hist_data)
  binSize <- range/numberOfBins
  farthest_number <- max(abs(hist_range)) # calculate the number farthest from 0
  x_min <- -(farthest_number)-2*(binSize) # set xmin so zero is centred
  x_max <- (farthest_number)+2*(binSize) # set xmax so zero is centred

  # get a subset of dataframe for the histogram
  plot_data <- data %>%
  mutate(!!sym(col_name) := as.numeric(!!sym(col_name))) %>%
  select(!!sym(col_name), !!sym(colourByCol)) %>%
  filter(!is.na(!!sym(col_name))) %>%
  arrange(!!sym(colourByCol))

  #View(plot_data)
  #str(plot_data)

```

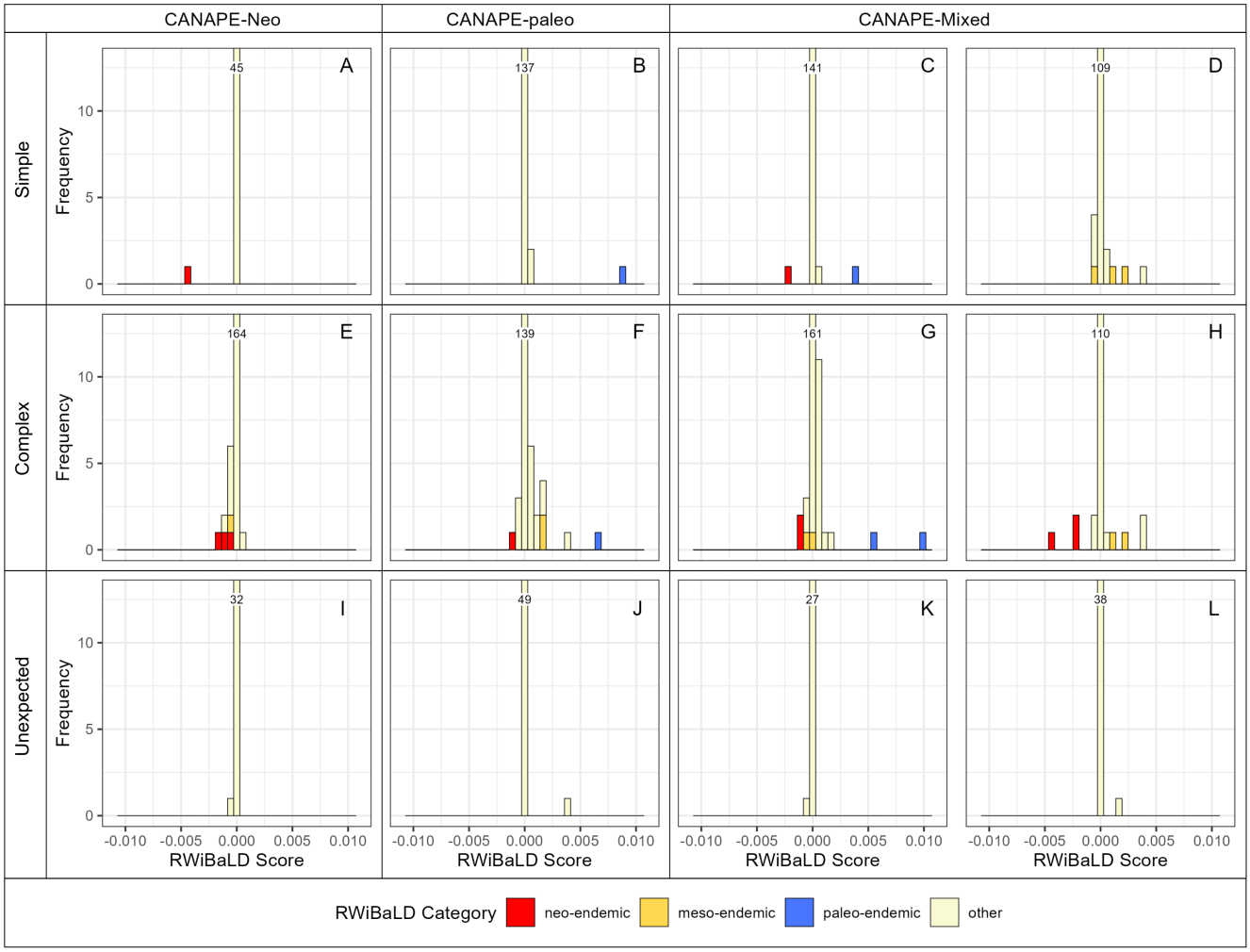
```
  plot <- ggplot(plot_data, aes(x = .data[[col_name]], fill = forcats::fct_rev(rwibald_type))) +
  geom_histogram(color = "black", linewidth = 0.2, bins = numberOfBins,  show.legend = showLegend) +
  scale_fill_manual(values = col_scheme, labels = legend_labels, breaks = legend_order, drop = FALSE,
   ↪  na.value = "transparent") +
  #xlim(x_min, x_max) +
  xlim(-0.011, 0.011) +
  coord_cartesian(ylim = c(0, 13)) +
  scale_y_continuous(labels = scales::label_number(accuracy = 1)) +
  labs(x = "RWiBaLD Score", y = "Frequency", fill = "RWiBaLD Category") +
  theme_bw()

 plot <- plot +
  geom_rect(
    aes(xmin = -0.0005, xmax = 0.0005, ymin = 12.2, ymax = 12.8),
    color = "white",
    fill = "white"
  ) +
   stat_bin(
    aes(label = after_stat(if_else (condition = count>12, as.character(count), ""))),
    bins= as.numeric(numberOfBins),
    position=position_stack(vjust=0.1),
    pad=TRUE,
    geom = "text",
    color = "black",
    size = 2.5,
    y = 12.5
  )

  #print(plot)

  # Dynamic variable name for the plot
  plot_name <- paste0("plot_", i - 1)  # Subtracting 1 to start numbering from 1
  assign(plot_name, plot)

  # Add the plot to the list
  plot_list[[i - 1]] <<- get(plot_name)
  #rm(plot)
  })

}

 #print(plot_list[[1]])

#save the plotlist for later on
saveRDS(plot_list, "quarto_outputs/plotlist.rds")
```

21 Generate multi-panel histogram figure 4A-L

This code compiles all 12 histograms generated above into one figure.

```
plot_list <- readRDS("quarto_outputs/plotlist.rds")

patchwork <-  plot_list[[1]] + xlab(NULL) +
              guides(x = "none") +
              plot_list[[2]] + xlab(NULL) + ylab(NULL) +
              guides(x = "none", y = "none") +
              plot_list[[3]] + xlab(NULL) + ylab(NULL) +
```

```
              guides(x = "none", y = "none") +
              plot_list[[4]] + xlab(NULL) + ylab(NULL) +
              guides(x = "none", y = "none") +
              plot_list[[5]] + xlab(NULL) + guides(x = "none") +
              plot_list[[6]] + xlab(NULL) + ylab(NULL) +
              guides(x = "none", y = "none") +
              plot_list[[7]] + xlab(NULL) + ylab(NULL) +
              guides(x = "none", y = "none") +
              plot_list[[8]] + xlab(NULL) + ylab(NULL) +
              guides(x = "none", y = "none") +
              plot_list[[9]] +
              plot_list[[10]] + ylab(NULL) + guides(y = "none") +
              plot_list[[11]] + ylab(NULL) + guides(y = "none") +
              plot_list[[12]] + ylab(NULL) + guides(y = "none") +
              plot_annotation(
              subtitle = '                    CANAPE-Neo                              CANAPE-paleo
               ↪  CANAPE-Mixed',
              caption = '',
              tag_levels = 'A') +
              plot_layout(ncol = 4, guides = 'collect') &
              theme(legend.position = "bottom", plot.tag.position = c(0.9, 0.9),
                    plot.tag = element_text(size = 12, hjust = 0, vjust = 0))

#print(patchwork)

ggsave("quarto_outputs/figures/Figure4_A.png", patchwork, width = 3000, height = 2400, units = "px")
```



22 Load results data & generate data files for biome maps

Now we take the RWiBaLD results and split the taxa in the original specimen data from the Categorical Analysis of Neo And Paleo-Endemisim (CANAPE) paper according to these RWiBaLD categories so we can overlay them on the biome map published by Crisp et al. in 2004.

```
original_paper_specimen_data_file <-
 ↪  "doi_10_5061_dryad_dv4qk__v20150514/Point_distribution_Australian_Phylogenetic_Diversity_Acacia.csv"

original_paper_specimen_data <- read.table(original_paper_specimen_data_file, header=T, sep=",")

#View(original_paper_specimen_data)
#View(rwibald_results_all_with_range)

# Summarize the data in the rwibaled_type column
summary_df <- rwibald_results_all_with_range %>%
  group_by(rwibald_type) %>%
  summarise(count = n())  # Count of each rwibaled_type

# Create a table using gt
summary_table <- summary_df %>%
  gt() %>%
```

Summary of rwibaled_type

Count of each type

rwibald_type	count
meso-endemic	59
neo-endemic	55
other	880
paleo-endemic	21

```
  tab_header(
    title = "Summary of rwibaled_type",
    subtitle = "Count of each type"
  )

# Print the table
summary_table

# Extract the 'NAME' values where 'rwibald_type' is 'meso-endemic'
meso_names <- rwibald_results_all_with_range[rwibald_results_all_with_range$rwibald_type ==
 ↪  "meso-endemic", "NAME"]

# Filter the 'original_paper_specimen_data' dataframe
meso_taxa <- original_paper_specimen_data[original_paper_specimen_data$Species %in% meso_names, ]

# Write data to file
write.csv(meso_taxa, "quarto_outputs/Meso_endemics.csv" ,row.names=FALSE)

# Extract the 'NAME' values where 'rwibald_type' is 'neo-endemic'
neo_names <- rwibald_results_all_with_range[rwibald_results_all_with_range$rwibald_type == "neo-endemic",
 ↪  "NAME"]

# Filter the 'original_paper_specimen_data' dataframe
neo_taxa <- original_paper_specimen_data[original_paper_specimen_data$Species %in% neo_names, ]

# Write data to file
write.csv(neo_taxa, "quarto_outputs/Neo_endemics.csv" ,row.names=FALSE)

# Extract the 'NAME' values where 'rwibald_type' is 'paleo-endemic'
paleo_names <- rwibald_results_all_with_range[rwibald_results_all_with_range$rwibald_type ==
 ↪  "paleo-endemic", "NAME"]

# Filter the 'original_paper_specimen_data' dataframe
paleo_taxa <- original_paper_specimen_data[original_paper_specimen_data$Species %in% paleo_names, ]

# Write data to file
write.csv(paleo_taxa, "quarto_outputs/Paleo_endemics.csv" ,row.names=FALSE)
```

23 Two tailed relative phylogenetic diversity (RPD) & CANAPE functions

This is the function for calculating Categorical Analysis of Neo- and Paleo-endemism (CANAPE) from ranked P scores of Biodiverse analysis.

```
#Standard 2 tailed test for RPD
significance_fun <- function(x){
```

```
  if (x >= 0.99) {
    return("Very Highly Sig")
  } else if (x >= 0.975){
    return ("Highly Sig")
  } else if (x <= 0.01){
    return ("Very Sig Low")
  } else if (x <= 0.025){
    return ("Sig Low")
  } else {
    return("Not Sig")
  }
}

#two pass test for RPE
# x=P_PE_WE_P, y=P_PHYLO_RPE_NULL2, z=P_PHYLO_RPE2
significance_super_fun <- function(x, y, z){
  if (x > 0.95 || y > 0.95) {
    if (z <= 0.025){
      return ("Neo")
    } else if (z >= 0.975){
      return ("paleo")
    } else if (x >= 0.99 || y >= 0.99){
      return ("Super")
    } else {
      return("Mixed")
    }
  } else {
    return("Not Sig")
  }
}
```

24 Load CANAPE & randomisation results

Here we load the re-created biodiverse results from the original paper, calculate CANAPE and re-generate the CANAPE map. Note: these results may differ slightly from the original paper as we re-ran the randomisation in biodiverse.

```
data_dir <- "Acacia_biodiverse_exports/"

# The CANAPE results file calculated in Biodiverse directly and exported
biodiverse_canape_results_file <- paste0(data_dir, "Acacia_Rand1_CANAPE_Export.csv")

# The spatial & randomisation results files calculated in Biodiverse and exported
biodiverse_observed_data_file  <- paste0(data_dir, "Acacia_SPATIAL_RESULTS_Export.csv")
biodiverse_rand_results_file <- paste0(data_dir, "Acacia_Rand1_SPATIAL_RESULTS_Export.csv")

biodiverse_canape_results <- read.table(biodiverse_canape_results_file, header=T,sep=",", check.names =
 ↪  FALSE )

biodiverse_observed_spatial_results <- read.table(biodiverse_observed_data_file, header=T,sep=",")
biodiverse_rand_spatial_results <- read.table(biodiverse_rand_results_file, header=T,sep=",")

biodiverse_results_concatenated <- cbind(biodiverse_observed_spatial_results,
 ↪  biodiverse_rand_spatial_results)

#View(biodiverse_results_concatenated)

################################################
```

```r
#Create new columns in dataframe and
#populate them using the functions above
################################################

targets <- c("PHYLO_RPD2", "PD_P", "PE_WE_P", "PD_P_per_taxon", "PHYLO_RPE2")

for (name in targets) {
  colname <- paste0("P_", name)  #  prepend the P_ since we want the proportions, saves some typing above
  new_colname = paste0(colname, "_SIG")
  trait_index <- match (colname, colnames(biodiverse_results_concatenated))
  # Apply the function to every row of column with index "trait_index"
  #  and generate a column in the dataframe showing significant cells
  if (!is.na(trait_index)) {
    biodiverse_results_concatenated[[new_colname]] <- apply
 ↪  (biodiverse_results_concatenated[trait_index],  MARGIN=c(1), significance_fun)
  } else {
    print (paste("Cannot find index", colname, "in data frame"))
  }
}

biodiverse_results_concatenated$P_PHYLO_RPE2_CANAPE_SIG <- sapply(
  1:nrow(biodiverse_results_concatenated),
  function(x) significance_super_fun(
    biodiverse_results_concatenated$P_PE_WE_P[x],
    biodiverse_results_concatenated$P_PHYLO_RPE_NULL2[x],
    biodiverse_results_concatenated$P_PHYLO_RPE2[x]
  )
)

#View(biodiverse_results_concatenated)
```

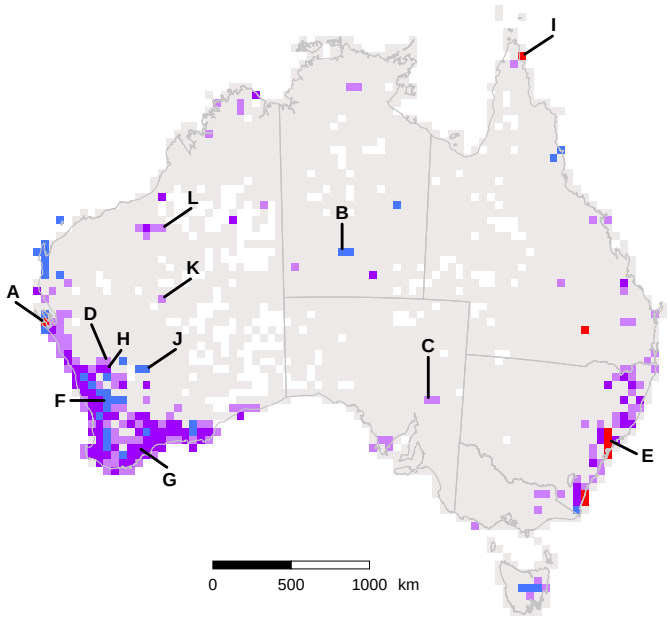
25 Generate CANAPE map figure 5A

Here we generate the CANAPE map, indicating cell locations from the histograms in figure 4A-L. CANAPE identifies geographic concentrations of high PE, and gives a summary classification of the type of endemism dominating in a location. The RWiBaLD histograms (figure 4A-L) identify the specific branches that contribute the most to PE in a given cell, and what type of endemism they represent.

```r
#loadfonts()

myFont <- choose_font(c("HelvLight", "Arial", "sans"), quiet = TRUE) #load a font if available

map_text <- "Categorical Analysis of Neo- And Paleo- Endemism"
sigplot <- "P_PHYLO_RPE2_CANAPE_SIG"
col_scheme <- c("paleo" = "royalblue1","Not Sig" = "snow2", "Neo" = "red", "Super" = "#9D00FF", "Mixed"=
 ↪  "#CB7FFF")
legend_order <-c("Neo","paleo", "Not Sig", "Mixed", "Super")
legend_labels <- c("Neo"="Neo","paleo"="Paleo", "Not Sig"="Not significant", "Mixed"="Mixed",
 ↪  "Super"="Super")

biodiverse_results_concatenated[, sigplot] <- factor(biodiverse_results_concatenated[, sigplot],
 ↪  levels=legend_order)
Axis_0 <- "Axis_0"
Axis_1 <- "Axis_1"

map_shape_file <- paste0("shape_files/coastline_albers.shp")
map_data   <- st_read(map_shape_file)

```

```r
cols_to_keep <- c("-1825000:-2975000", "75000:-2525000",
 ↪  "625000:-3475000","-1425000:-3225000","1775000:-3725000", "-1425000:-3475000", "-1225000:-3775000",
 ↪  "-1425000:-3275000", "1225000:-1275000", "-1175000:-3275000", "-1075000:-2825000",
 ↪  "-1075000:-2375000")

# Initialize empty vectors for x and y
x <- c()
y <- c()

# Split each element of cols_to_keep and append to x and y for plotting on the map
for (val in cols_to_keep) {
  parts <- strsplit(val, ":")[[1]]
  x <- c(x, as.numeric(parts[1]))
  y <- c(y, as.numeric(parts[2]))
}

# Print the result
#print(x)
#print(y)

labels <- c("A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L")

# Create data frame
histograms <- data.frame(x = x, y = y, label = labels)

# Plot the sf object with ggplot2, coloring by the BIOME column
map_plot_CANAPE  <- ggplot() +
  geom_tile(data=biodiverse_results_concatenated, aes_string(x=Axis_0, y=Axis_1, fill=sigplot)) +
  geom_sf(data = map_data, colour = "grey77" , fill="transparent") +
  scale_fill_manual(values = col_scheme,  labels=legend_labels, name="CANAPE", guide =
   ↪  guide_legend(direction = "horizontal", title.position = "bottom", title.hjust=0.5, title.vjust=0.5,
   ↪  label.position="bottom", label.hjust = 0.5, label.vjust = 0.1, lineheight=0.5))+
  geom_text_repel(data=histograms, aes(x = x, y = y, label = labels), fontface = "bold", size = 2.5,
   ↪  nudge_x = c(-200000, 0, 0, -100000, 250000, -300000, 200000, 100000, 200000, 200000, 200000,
   ↪  200000), nudge_y = c(200000, 250000, 350000, 300000, -100000, 0, -200000, 200000, 200000, 200000,
   ↪  200000, 200000)) +
   annotate("rect", xmin = -750000, xmax = -250000, ymin = -4500000, ymax = -4550000, fill = "black",
    ↪  colour = "black", alpha = 1, linewidth = 0.1) +
    annotate("rect", xmin = -250000, xmax = 250000, ymin = -4500000, ymax = -4550000, fill = "white",
     ↪  colour = "black", alpha = 1, linewidth = 0.1) +
    annotate("text", label = "0", x = -750000, y = -4650000, size=rel(2),  face = 'plain', family =
     ↪  myFont) +
    annotate("text", label = "500", x = -250000, y = -4650000, size=rel(2),  face = 'plain', family =
     ↪  myFont) +
    annotate("text", label = "1000", x = 250000, y = -4650000, size=rel(2),  face = 'plain', family =
     ↪  myFont) +
    annotate("text", label = "km", x = 500000, y = -4650000, size=rel(2),  face = 'plain', family =
     ↪  myFont) +
   theme(text = element_text(family = myFont),
         strip.background = element_blank(),
         title = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family = myFont),
         axis.line=element_blank(),axis.text.x=element_blank(),
         axis.text.y=element_blank(),axis.ticks=element_blank(),
         axis.title.x=element_blank(), axis.title.y=element_blank(),
         legend.position="none",
         legend.direction='horizontal',
         legend.text = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family =
          ↪  myFont),
         panel.grid = element_blank(),
```

```
        panel.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
        panel.border = element_blank(),
        plot.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
        plot.margin=unit(c(0,0,0,0),"line"))

print(map_plot_CANAPE)
```




```
ggsave('quarto_outputs/figures/Figure5_A.png', map_plot_CANAPE, width = 1024, height = 1024, units =
↪  "px", bg = "white")
```

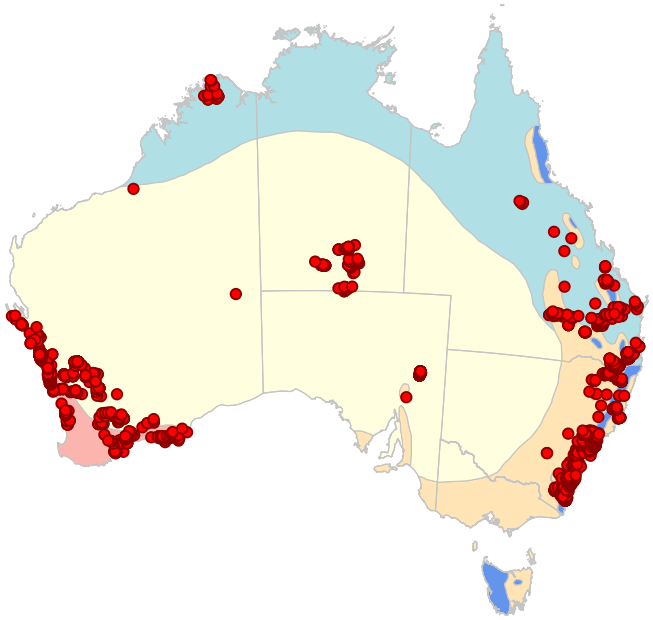
```
Reading layer `coastline_albers' from data source
  `C:\E\gitRepos\rwibald\shape_files\coastline_albers.shp' using driver `ESRI Shapefile'
Simple feature collection with 44 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -1887741 ymin: -4840771 xmax: 2121462 ymax: -1029671
Projected CRS: GDA94 / Australian Albers
```

26 Generate biome map with neo-endemics figure 5B

Here we show the specimen data for all of the neo-endemic taxa categorised by RWiBaLD overlaid on the biome map published by Crisp et al. in 2004.

```
#loadfonts()

#myFont <- choose_font(c("HelvLight", "Arial", "sans"), quiet = TRUE) #load a font if available

col_scheme <- c("aseasonal wet" = "cornflowerblue",
                "eremean" = "lightyellow1",
                "monsoonal tropics" = "powderblue",
                "southeastern temperate" = "moccasin",
                "southwestern temperate"= "#fbb4ae")
```

```r
legend_order <-c("aseasonal wet",
                 "eremean",
                 "monsoonal tropics",
                 "southeastern temperate",
                 "southwestern temperate")
legend_labels <- c("aseasonal wet" = "aseasonal wet",
                   "eremean" = "eremean",
                   "monsoonal tropics" = "monsoonal tropics",
                   "southeastern temperate"="southeastern temperate",
                   "southwestern temperate"="southwestern temperate")

map_shape_file1 <- paste0("shape_files/coastline_albers.shp")
map_data   <- st_read(map_shape_file1)
#geom_sf(data = map_data, colour = "grey77" , fill="transparent") +

map_shape_file <- paste0("shape_files/biomes_crisp_3577/biomes_crisp_3577.shp")
# Read the shapefile as an sf object
biomes_sf <- st_read(map_shape_file)

csv_file <- paste0("quarto_outputs/Neo_endemics.csv")
# Read the CSV file into a data frame
csv_data <- read.csv(csv_file)
# Convert data frame to sf object
csv_sf <- st_as_sf(csv_data, coords = c("x_metres_EPSG_3577_Albers_Equal_Area",
 ↪  "y_metres_EPSG_3577_Albers_Equal_Area"), crs = st_crs(biomes_sf))


# Plot the sf object with ggplot2, coloring by the BIOME column
map_plot_Neo  <- ggplot() +
  geom_sf(data = biomes_sf, aes(fill = BIOME), colour = "grey77") +
  geom_sf(data = map_data, colour = "grey77" , fill="transparent") +
  geom_sf(data = csv_sf, fill= "red", color = "red4", shape = 21) +
 scale_fill_manual(values = col_scheme,  labels=legend_labels, name="Biomes", guide =
  ↪  guide_legend(direction = "horizontal", title.position = "bottom", title.hjust=0.5, title.vjust=0.5,
  ↪  label.position="bottom", label.hjust = 0.5, label.vjust = 0.1, lineheight=0.5))+
   theme(text = element_text(family = myFont),
         strip.background = element_blank(),
         title = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family = myFont),
         axis.line=element_blank(),axis.text.x=element_blank(),
         axis.text.y=element_blank(),axis.ticks=element_blank(),
         axis.title.x=element_blank(), axis.title.y=element_blank(),
         legend.position="none",
         legend.direction='horizontal',
         legend.text = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family =
          ↪  myFont),
         legend.key = element_rect(color = "black", size = 0),
         panel.grid = element_blank(),
         panel.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
         panel.border = element_blank(),
         plot.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
         plot.margin=unit(c(0,0,0,0),"line"))


print(map_plot_Neo)
```



```
ggsave("quarto_outputs/figures/Figure5_B.png", map_plot_Neo, width = 3000, height = 2600, units = "px")
```

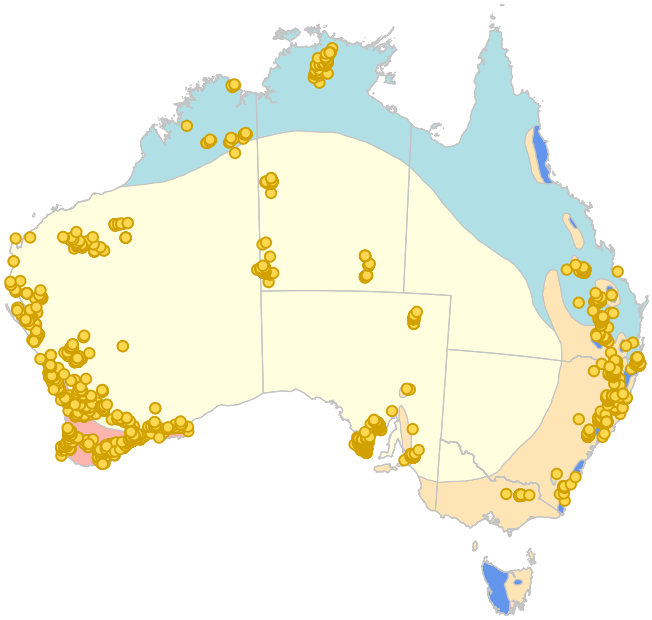
```
Reading layer `coastline_albers' from data source
  `C:\E\gitRepos\rwibald\shape_files\coastline_albers.shp' using driver `ESRI Shapefile'
Simple feature collection with 44 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -1887741 ymin: -4840771 xmax: 2121462 ymax: -1029671
Projected CRS: GDA94 / Australian Albers
Reading layer `biomes_crisp_3577' from data source
  `C:\E\gitRepos\rwibald\shape_files\biomes_crisp_3577\biomes_crisp_3577.shp'
  using driver `ESRI Shapefile'
Simple feature collection with 5 features and 3 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -1887414 ymin: -4840588 xmax: 2121442 ymax: -1087816
Projected CRS: GDA94 / Australian Albers
```

27 Generate biome map with meso-endemics figure 5C

Here we show the specimen data for all of the meso-endemic taxa categorised by RWiBaLD overlaid on the biome map published by Crisp et al. in 2004.

```
#loadfonts()

#myFont <- choose_font(c("HelvLight", "Arial", "sans"), quiet = TRUE) #load a font if available

map_text <- "Categorical Analysis of Neo- And Paleo- Endemism"

col_scheme <- c("aseasonal wet" = "cornflowerblue",
                "eremean" = "lightyellow1",
                "monsoonal tropics" = "powderblue",
                "southeastern temperate" = "moccasin",
```

```
                "southwestern temperate"= "#fbb4ae")
legend_order <-c("aseasonal wet",
                 "eremean",
                 "monsoonal tropics",
                 "southeastern temperate",
                 "southwestern temperate")
legend_labels <- c("aseasonal wet" = "aseasonal wet",
                   "eremean" = "eremean",
                   "monsoonal tropics" = "monsoonal tropics",
                   "southeastern temperate"="southeastern temperate",
                   "southwestern temperate"="southwestern temperate")

map_shape_file1 <- paste0("shape_files/coastline_albers.shp")
map_data   <- st_read(map_shape_file1)
#geom_sf(data = map_data, colour = "grey77" , fill="transparent") +

map_shape_file <- paste0("shape_files/biomes_crisp_3577/biomes_crisp_3577.shp")
# Read the shapefile as an sf object
biomes_sf <- st_read(map_shape_file)

csv_file <- paste0("quarto_outputs/Meso_endemics.csv")
# Read the CSV file into a data frame
csv_data <- read.csv(csv_file)
# Convert data frame to sf object
csv_sf <- st_as_sf(csv_data, coords = c("x_metres_EPSG_3577_Albers_Equal_Area",
 ↪  "y_metres_EPSG_3577_Albers_Equal_Area"), crs = st_crs(biomes_sf))


# Plot the sf object with ggplot2, coloring by the BIOME column
map_plot_Meso  <- ggplot() +
  geom_sf(data = biomes_sf, aes(fill = BIOME), color = "grey77") +
  geom_sf(data = map_data, colour = "grey77" , fill="transparent") +
  geom_sf(data = csv_sf, fill= "#FFD851", color = "#D0A100", shape = 21) +
  scale_fill_manual(values = col_scheme,  labels=legend_labels, name="Biomes", guide =
   ↪  guide_legend(direction = "horizontal", title.position = "bottom", title.hjust=0.5, title.vjust=0.5,
   ↪  label.position="bottom", label.hjust = 0.5, label.vjust = 0.1, lineheight=0.5))+
   theme(text = element_text(family = myFont),
         strip.background = element_blank(),
         title = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family = myFont),
         axis.line=element_blank(),axis.text.x=element_blank(),
         axis.text.y=element_blank(),axis.ticks=element_blank(),
         axis.title.x=element_blank(), axis.title.y=element_blank(),
         legend.position="none",
         legend.direction='horizontal',
         legend.text = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family =
          ↪  myFont),
         legend.key = element_rect(color = "black", size = 0),
         panel.grid = element_blank(),
         panel.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
         panel.border = element_blank(),
         plot.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
         plot.margin=unit(c(0,0,0,0),"line"))

print(map_plot_Meso)
```



```
#cvdPlot(map_plot_Meso)

ggsave("quarto_outputs/figures/Figure5_C.png", map_plot_Meso, width = 3000, height = 2600, units = "px")
```

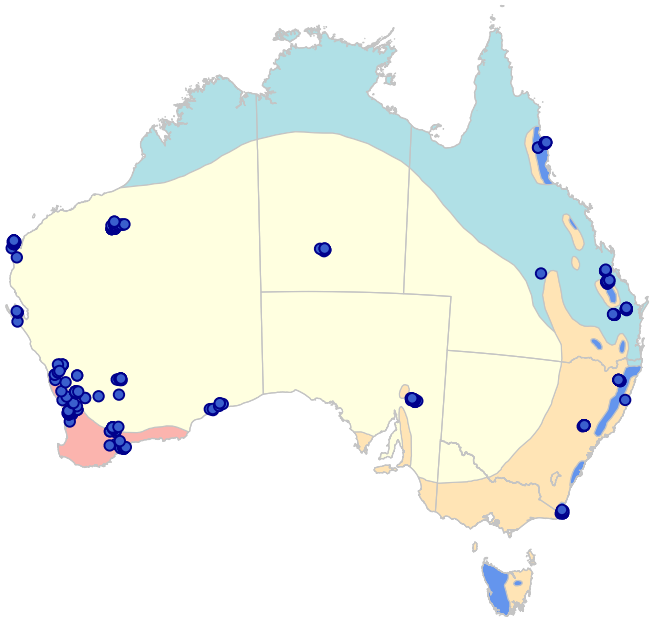
```
Reading layer `coastline_albers' from data source
  `C:\E\gitRepos\rwibald\shape_files\coastline_albers.shp' using driver `ESRI Shapefile'
Simple feature collection with 44 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -1887741 ymin: -4840771 xmax: 2121462 ymax: -1029671
Projected CRS: GDA94 / Australian Albers
Reading layer `biomes_crisp_3577' from data source
  `C:\E\gitRepos\rwibald\shape_files\biomes_crisp_3577\biomes_crisp_3577.shp'
  using driver `ESRI Shapefile'
Simple feature collection with 5 features and 3 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -1887414 ymin: -4840588 xmax: 2121442 ymax: -1087816
Projected CRS: GDA94 / Australian Albers
```

28 Generate biome map with paleo-endemics figure 5D

Here we show the specimen data for all of the paleo-endemic taxa categorised by RWiBaLD overlaid on the biome map published by Crisp et al. in 2004.

```
#loadfonts()

#myFont <- choose_font(c("HelvLight", "Arial", "sans"), quiet = TRUE) #load a font if available

map_text <- "Categorical Analysis of Neo- And Paleo- Endemism"

col_scheme <- c("aseasonal wet" = "cornflowerblue",
                "eremean" = "lightyellow1",
```

```
                "monsoonal tropics" = "powderblue",
                "southeastern temperate" = "moccasin",
                "southwestern temperate"= "#fbb4ae")
legend_order <-c("aseasonal wet",
                 "eremean",
                 "monsoonal tropics",
                 "southeastern temperate",
                 "southwestern temperate")
legend_labels <- c("aseasonal wet" = "aseasonal wet",
                   "eremean" = "eremean",
                   "monsoonal tropics" = "monsoonal tropics",
                   "southeastern temperate"="southeastern temperate",
                   "southwestern temperate"="southwestern temperate")

map_shape_file1 <- paste0("shape_files/coastline_albers.shp")
map_data  <- st_read(map_shape_file1)
#geom_sf(data = map_data, colour = "grey77" , fill="transparent") +

map_shape_file <- paste0("shape_files/biomes_crisp_3577/biomes_crisp_3577.shp")

# Read the shapefile as an sf object
biomes_sf <- st_read(map_shape_file)

csv_file <- paste0("quarto_outputs/Paleo_endemics.csv")

# Read the CSV file into a data frame
csv_data <- read.csv(csv_file)

# Convert data frame to sf object
csv_sf <- st_as_sf(csv_data, coords = c("x_metres_EPSG_3577_Albers_Equal_Area",
 ↪  "y_metres_EPSG_3577_Albers_Equal_Area"), crs = st_crs(biomes_sf))

# Plot the sf object with ggplot2, coloring by the BIOME column
map_plot_Paleo  <- ggplot() +
  geom_sf(data = biomes_sf, aes(fill = BIOME), color = "grey77") +
  geom_sf(data = map_data, colour = "grey77" , fill="transparent") +
  geom_sf(data = csv_sf, fill= "royalblue3", color = "darkblue", shape = 21) +
  scale_fill_manual(values = col_scheme,  labels=legend_labels, name="Biomes", guide =
   ↪  guide_legend(direction = "horizontal", title.position = "bottom", title.hjust=0.5, title.vjust=0.5,
   ↪  label.position="bottom", label.hjust = 0.5, label.vjust = 0.1, lineheight=0.5))+
    theme(text = element_text(family = myFont),
         strip.background = element_blank(),
         title = element_text(colour = 'black', angle = 0, size=rel(1), face = 'plain', family = myFont),
         axis.line=element_blank(),axis.text.x=element_blank(),
         axis.text.y=element_blank(),axis.ticks=element_blank(),
         axis.title.x=element_blank(), axis.title.y=element_blank(),
         legend.position="none",
         legend.direction='horizontal',
         legend.text = element_text(colour = 'black', angle = 0, size=rel(0.5), face = 'plain', family =
          ↪  myFont),
         legend.key = element_rect(color = "black", size = 0),
         panel.grid = element_blank(),
         panel.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
         panel.border = element_blank(),
         plot.background=element_blank(),#element_rect(colour = "black", fill="white", size = 1),
         plot.margin=unit(c(0,0,0,0),"line"))


print(map_plot_Paleo)
```



```
ggsave("quarto_outputs/figures/Figure5_D.png", map_plot_Paleo, width = 3000, height = 2600, units = "px")
```

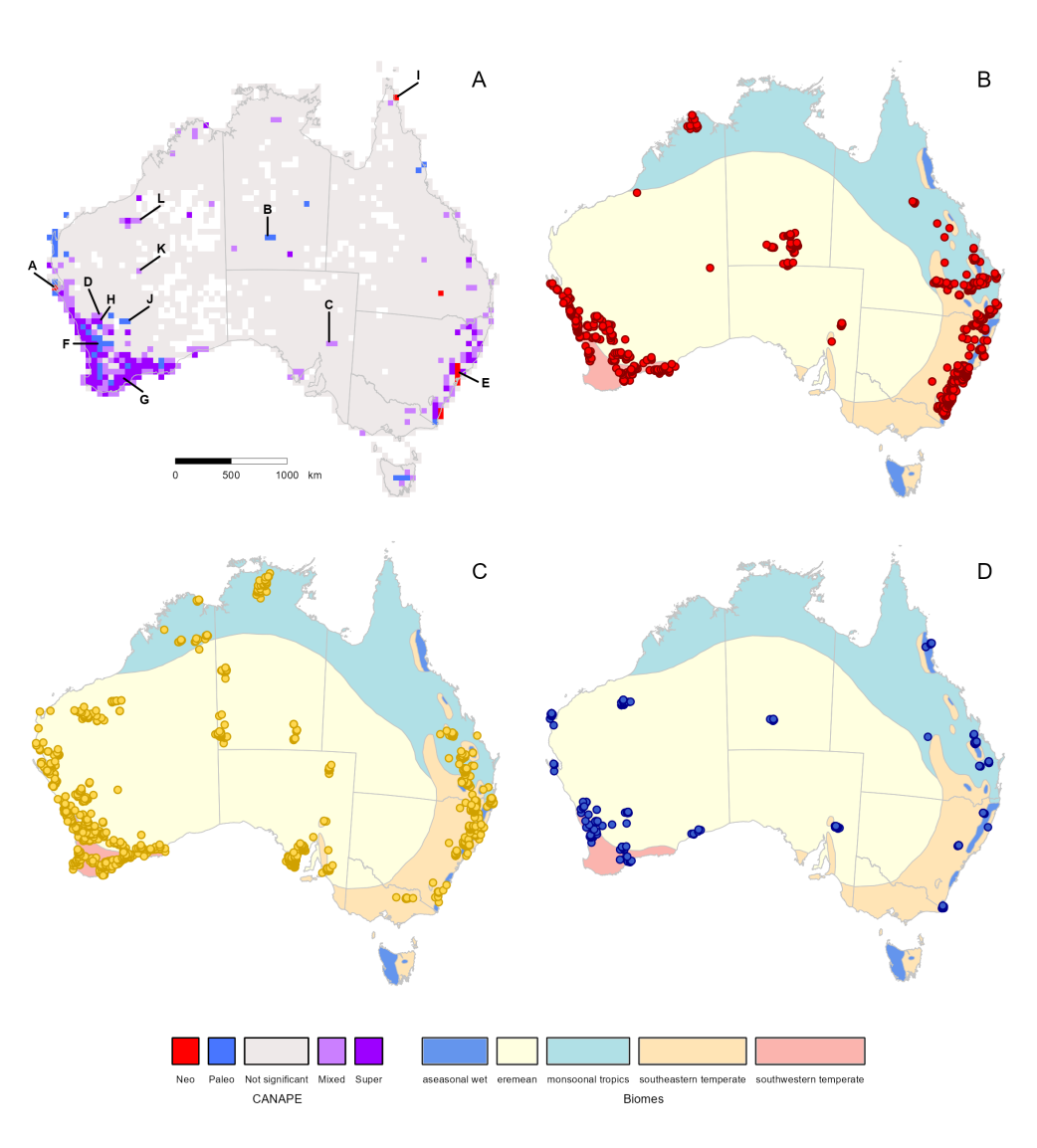
```
Reading layer `coastline_albers' from data source
  `C:\E\gitRepos\rwibald\shape_files\coastline_albers.shp' using driver `ESRI Shapefile'
Simple feature collection with 44 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -1887741 ymin: -4840771 xmax: 2121462 ymax: -1029671
Projected CRS: GDA94 / Australian Albers
Reading layer `biomes_crisp_3577' from data source
  `C:\E\gitRepos\rwibald\shape_files\biomes_crisp_3577\biomes_crisp_3577.shp'
  using driver `ESRI Shapefile'
Simple feature collection with 5 features and 3 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -1887414 ymin: -4840588 xmax: 2121442 ymax: -1087816
Projected CRS: GDA94 / Australian Albers
```

29 Generate 4 up map figure 5ABCD

Here we compile the maps into one figure 5A-D

```
patchwork_map <- map_plot_CANAPE +
              map_plot_Neo +
              map_plot_Meso +
              map_plot_Paleo

patchwork_map <- patchwork_map +
              plot_annotation(subtitle = '',
                              caption = '',
                              tag_levels = 'A') +
```

```
              plot_layout(ncol = 2, guides = 'collect') &
              theme(legend.position = "bottom",
                    legend.title=element_text(size=rel(0.6)),
                    legend.text=element_text(size=rel(0.5)),
                    legend.title.position = "bottom",
                    legend.label.position= "bottom",
                    legend.key = element_rect(color = "black", size = 0.5),
                    legend.key.height = unit(0.5, "cm"),
                    legend.key.width = unit(0.5, "cm"),
                    plot.tag.position = c(0.9, 0.9),
                    plot.tag = element_text(size = 12, hjust = 0, vjust = 0))

#print(patchwork_map)

#cvdPlot(patchwork_map)

ggsave('quarto_outputs/figures/Figure5_ABCD.png', patchwork_map, width = 3000, height = 2600, units =
 ↪  "px")
```



30 Table of RWiBaLD results supplementary table 1

RWiBaLD results for each branch on the phylogeny of Australian Acacia.

```
RWiBaLD_results_csv  <- paste0("quarto_outputs/Acacia_RWiBaLD_results_all_with_range.csv")

rwibald_results_all_with_range <- read.table(RWiBaLD_results_csv, header=T,sep=",")

#View(rwibald_results_all_with_range)

#colnames(rwibald_results_all_with_range)

gt_table <- rwibald_results_all_with_range %>%
            select("NAME", "branch_length_comparison_tree", "branch_length_observed_tree",
                   "rwibald_score", "rwibald_type", "range_cell_count")

# ---- Table output: gt for HTML/PDF, kable for GFM ----
```

```r
if (knitr::is_html_output() || knitr::is_latex_output()) {

  rwibald_results_all_with_range_table <- gt::gt(gt_table) |>
    gt::tab_options(
      table.width = gt::pct(100),
      table.layout = "auto",
      table.align = "left",
      table.margin.left = gt::px(5),
      table.margin.right = gt::px(5),
      table.font.size = gt::px(8),
      column_labels.font.size = gt::px(10),
      heading.align = "center",
      heading.title.font.size = gt::px(12),
      quarto.use_bootstrap = TRUE
    ) |>
    gt::opt_row_striping() |>
    gt::tab_header(title = gt::md("RWiBaLD Results")) |>
    gt::tab_style(
      style = gt::cell_text(align = "center"),
      locations = gt::cells_body()
    )

  rwibald_results_all_with_range_table

} else {

  # GFM-safe fallback (pipe table)
  knitr::kable(
    gt_table,
    format = "pipe",
    align  = "c"
  )

}


# print table using gt
# rwibald_results_all_with_range_table <- gt(gt_table) %>%
#   tab_options(
#     table.width = pct(100),
#     table.layout = "auto",
#     table.align = "left",
#     table.margin.left = px(5),
#     table.margin.right = px(5),
#     table.font.size = px(8),
#     column_labels.font.size = px(10),
#     heading.align = "center",
#     heading.title.font.size = px(12),
#     quarto.use_bootstrap = TRUE
#   ) %>%
#   opt_row_striping() %>%
#   tab_header(
#     title = md("RWiBaLD Results")
#   ) %>%
#   tab_style(
#     style = cell_text(align = "center"),
#     locations = cells_body()
#   )

```

```
#rwibald_results_all_with_range_table
```

RWiBaLD Results

NAME	branch_length_comparison_tree	branch_length_observed_tree	rwibald_score	rwibald_type	range_cell_count
1___	8.777963e-05	2.234812e-04	1.357015e-04	other	103
10___	8.951784e-05	3.583689e-05	-5.368095e-05	other	101
100___	4.109683e-05	7.239508e-06	-3.385732e-05	other	220
101___	2.366833e-05	3.361011e-05	9.941780e-06	other	382
102___	2.318283e-05	1.884262e-05	-4.340207e-06	other	390
103___	2.137424e-05	4.697007e-05	2.559583e-05	other	423
104___	2.318283e-04	9.137027e-04	6.818744e-04	other	39
105___	6.027535e-04	1.624921e-03	1.022168e-03	other	15
106___	1.532424e-04	1.057493e-04	-4.749313e-05	other	59
107___	9.935497e-05	3.545809e-04	2.552259e-04	other	91
108___	9.827502e-05	3.429999e-05	-6.397503e-05	other	92
109___	8.449815e-05	1.130098e-04	2.851166e-05	other	107
11___	8.777963e-05	2.896118e-05	-5.881846e-05	other	103
110___	1.891486e-05	9.029480e-06	-9.885378e-06	other	478
111___	7.534419e-04	1.343083e-03	5.896411e-04	other	12
112___	1.614518e-04	4.342113e-05	-1.180307e-04	other	56
113___	1.051314e-04	1.208033e-04	1.567189e-05	other	86
114___	8.777963e-05	6.943275e-09	-8.777269e-05	other	103
115___	7.862002e-05	9.885975e-06	-6.873404e-05	other	115
116___	7.175637e-05	1.346953e-05	-5.828684e-05	other	126
117___	7.008761e-05	4.083340e-05	-2.925421e-05	other	129
118___	7.008761e-05	5.616836e-05	-1.391925e-05	other	129
119___	6.954848e-05	1.778353e-05	-5.176495e-05	other	130
12___	1.335495e-05	1.097784e-05	-2.377112e-06	other	677
120___	1.597403e-05	7.837726e-06	-8.136307e-06	other	566
121___	1.504376e-05	1.467145e-05	-3.723155e-07	other	601
122___	9.881205e-06	4.100078e-06	-5.781126e-06	other	915
123___	9.859654e-06	1.414163e-06	-8.445490e-06	other	917
124___	3.117690e-04	3.328489e-04	2.107982e-05	other	29
125___	2.511473e-04	3.887985e-04	1.376512e-04	other	36
126___	2.054841e-04	4.573766e-05	-1.597465e-04	other	44
127___	2.825407e-04	6.909660e-05	-2.134441e-04	other	32
128___	2.825407e-04	2.411957e-04	-4.134496e-05	other	32
129___	5.022946e-04	2.914361e-04	-2.108585e-04	other	18
13___	1.310334e-05	5.646846e-06	-7.456490e-06	other	690
130___	1.130163e-04	3.901460e-05	-7.400168e-05	other	80
131___	1.004589e-04	4.157142e-05	-5.888750e-05	other	90
132___	7.862002e-05	1.278658e-04	4.924578e-05	other	115
133___	7.662121e-05	3.954025e-05	-3.708095e-05	other	118
134___	3.863804e-05	1.794236e-06	-3.684381e-05	other	234
135___	4.346780e-05	4.264787e-05	-8.199310e-07	other	208
136___	6.551668e-05	4.443585e-05	-2.108084e-05	other	138
137___	2.807858e-05	5.238707e-05	2.430849e-05	other	322
138___	2.539692e-05	7.418464e-05	4.878772e-05	other	356
139___	4.758580e-04	6.521900e-05	-4.106390e-04	other	19
14___	1.145919e-05	4.368943e-06	-7.090249e-06	other	789
140___	3.767209e-04	2.539890e-04	-1.227320e-04	other	24
141___	3.477424e-04	5.308590e-04	1.831166e-04	other	26
142___	2.916549e-04	4.352484e-04	1.435935e-04	other	31
143___	2.054841e-04	2.656276e-05	-1.789214e-04	other	44
144___	1.506884e-04	1.191929e-08	-1.506765e-04	other	60
145___	1.238535e-04	5.947281e-05	-6.438065e-05	other	73
146___	1.221798e-04	3.874051e-05	-8.343925e-05	other	74
147___	8.145317e-05	9.120355e-05	9.750379e-06	other	111
148___	1.936039e-05	8.736721e-06	-1.062367e-05	other	467
149___	1.361642e-05	7.233721e-06	-6.382699e-06	other	664
15___	1.090628e-05	2.498268e-05	1.407641e-05	other	829
150___	1.302781e-05	3.732148e-06	-9.295665e-06	other	694
151___	3.931001e-04	1.185664e-04	-2.745337e-04	other	23
152___	2.318283e-04	6.839497e-05	-1.634333e-04	other	39
153___	1.586193e-04	1.557460e-04	-2.873300e-06	other	57
154___	7.008761e-05	6.309607e-06	-6.377801e-05	other	129
155___	6.551668e-05	1.174379e-05	-5.377289e-05	other	138
156___	6.504534e-05	4.422731e-05	-2.081803e-05	other	139
157___	3.631045e-05	1.614817e-05	-2.016228e-05	other	249
158___	5.758791e-05	5.237361e-05	-5.214300e-06	other	157
159___	4.589494e-05	1.856155e-05	-2.733338e-05	other	197
16___	6.901757e-05	9.388402e-05	2.486645e-05	other	131
160___	4.000576e-05	2.116269e-05	-1.884307e-05	other	226
161___	2.620667e-05	2.072920e-09	-2.620460e-05	other	345
162___	1.369894e-04	5.770269e-05	-7.928674e-05	other	66
163___	1.027421e-04	1.694995e-05	-8.579212e-05	other	88
164___	8.001152e-05	3.946034e-05	-4.055118e-05	other	113
165___	5.318413e-05	4.206808e-09	-5.317992e-05	other	170
166___	4.224908e-05	3.435192e-06	-3.881388e-05	other	214
167___	9.132629e-05	2.496945e-05	-6.635684e-05	other	99
168___	2.825407e-04	4.848574e-05	-2.340550e-04	other	32
169___	1.674315e-04	2.781283e-05	-1.396187e-04	other	54
17___	6.192673e-05	8.906534e-05	2.713862e-05	other	146
170___	1.205507e-04	3.798806e-05	-8.256264e-05	other	75
171___	1.205507e-04	5.692448e-05	-6.362621e-05	other	75
172___	1.205507e-04	1.042297e-04	-1.632098e-05	other	75
173___	1.174195e-04	4.595375e-05	-7.146576e-05	other	77
174___	1.063683e-04	1.635680e-06	-1.047326e-04	other	85
175___	8.777963e-05	6.665401e-06	-8.111423e-05	other	103
176___	5.079383e-05	4.017738e-09	-5.078982e-05	other	178
177___	4.967749e-05	1.645418e-05	-3.322331e-05	other	182
178___	3.631045e-05	2.872118e-09	-3.630758e-05	other	249
179___	2.816605e-05	1.232424e-05	-1.584181e-05	other	321
18___	2.379290e-04	2.770606e-05	-2.102229e-04	other	38
180___	2.226922e-05	2.067711e-06	-2.020151e-05	other	406
181___	1.583415e-05	9.804553e-06	-6.029602e-06	other	571
182___	1.382462e-05	2.249130e-06	-1.157549e-05	other	654
183___	5.795707e-05	6.489423e-05	6.899068e-06	other	156
184___	5.413953e-05	3.121126e-05	-2.292827e-05	other	167
185___	8.371576e-05	6.061842e-05	-2.309734e-05	other	108
186___	8.072591e-05	6.385333e-09	-8.071953e-05	other	112
187___	7.930967e-05	2.326218e-05	-5.604749e-05	other	114