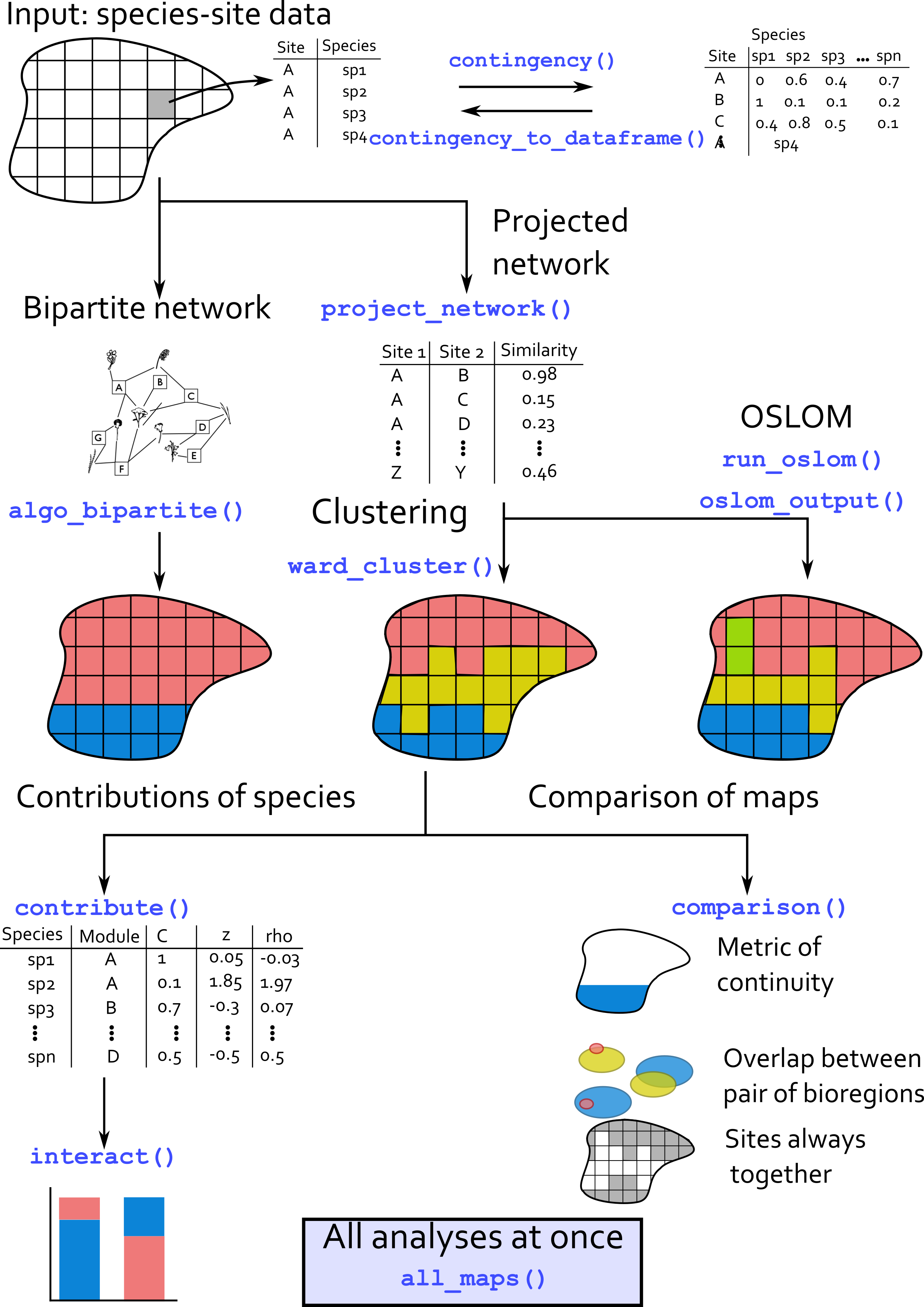
Tutorial for Bioregionalization R package

Pierre Denelle

2020-01-30

The principle of the package is illustrated by the following figure.

knitr::include\_graphics("../figures/Bioregionalization\_scheme3.png")



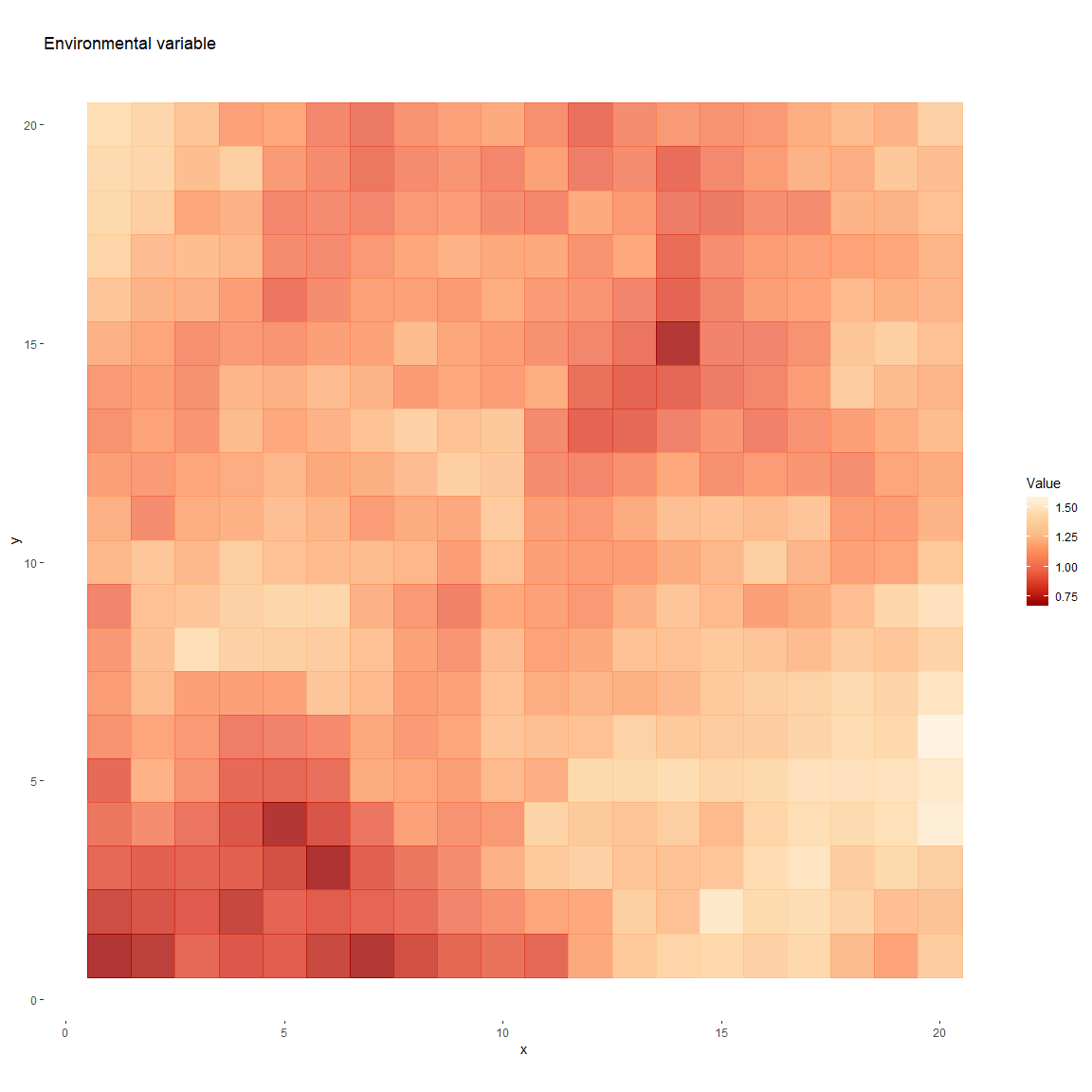
# Virtual dataset

virtual\_sp is a dataset simulated that comes with the package. This dataset relies on the response curve of virtual species to a virtual raster. The virtual raster contains 10000 cells and was simulated using gstat R package. [See here for details.](http://santiago.begueria.es/2010/10/generating-spatially-correlated-random-fields-with-r/)

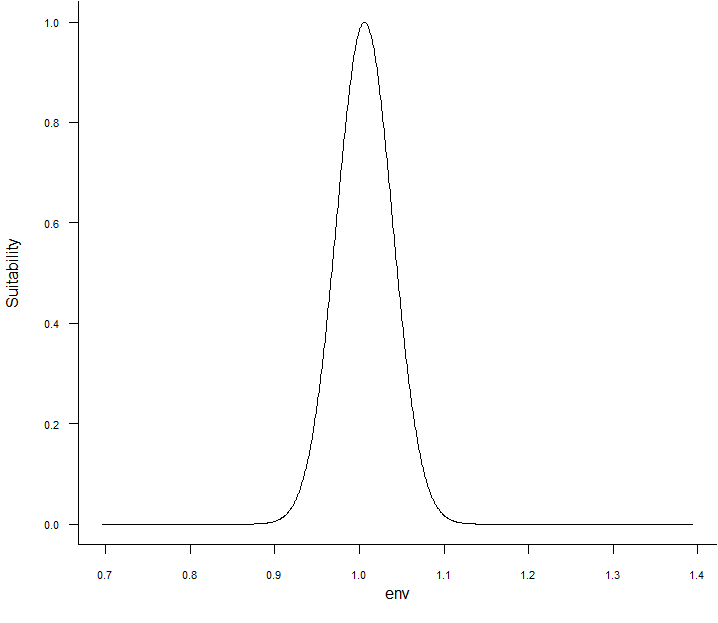
Based on this layer, the virtualspecies R package (Leroy et al. 2015) was used to simulate a Gaussian response curve of 100 virtual species. The mean and standard deviation of the response function was varying among species, such as some of them are more or less generalists/specialists.

For every species in every cell, we could derive a suitability index. Species with suitability index inferior to 0.15 were arbitrarily set absent.

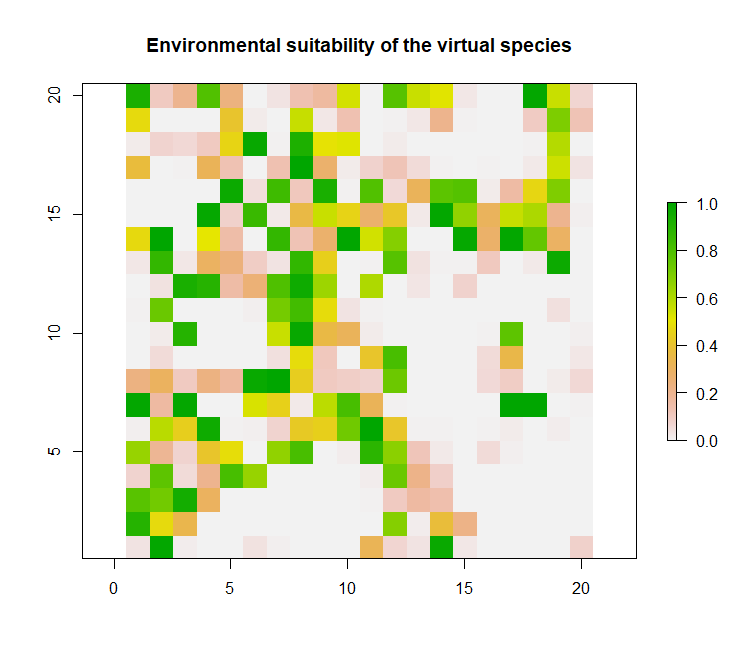
# Import virtual dataset  
data("virtual\_sp")  
  
# Only species data.frame  
sp\_df <- virtual[[1]]  
  
# Plot of environmental values  
sp\_df %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = env, color = env),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_distiller("Value", palette = "OrRd") +  
 scale\_fill\_distiller("Value", palette = "OrRd") +  
 coord\_equal() +  
 labs(title = "Environmental variable") +  
 theme(panel.background = element\_rect(fill = "transparent",colour = NA))



knitr::include\_graphics("../figures/sp50\_response\_curve.png")



knitr::include\_graphics("../figures/sp50\_suitability.png")



The first step is to convert the data.frame into a contingency table.

sp\_mat <- contingency(sp\_df[which(sp\_df$pa != 0), ],  
 "site", "sp", ab = NULL, binary = TRUE)  
knitr::kable(sp\_mat[1:5, 1:5])

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | sp1 | sp2 | sp3 | sp4 | sp5 |
| site1 | 1 | 1 | 1 | 1 | 1 |
| site10 | 0 | 0 | 0 | 0 | 0 |
| site100 | 0 | 0 | 0 | 0 | 0 |
| site101 | 0 | 0 | 0 | 0 | 0 |
| site102 | 0 | 0 | 0 | 0 | 0 |

# With weights  
sp\_mat\_w <- contingency(sp\_df, "site", "sp", ab = "suitab", binary = FALSE)  
knitr::kable(sp\_mat\_w[1:5, 1:5])

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | sp1 | sp2 | sp3 | sp4 | sp5 |
| site1 | 0.9158657 | 0.9312874 | 0.9428647 | 0.9517664 | 0.9587522 |
| site10 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| site100 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| site101 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |
| site102 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 | 0.0000000 |

# Community detection on projected networks

## Projection of the network

We then need to project the network.

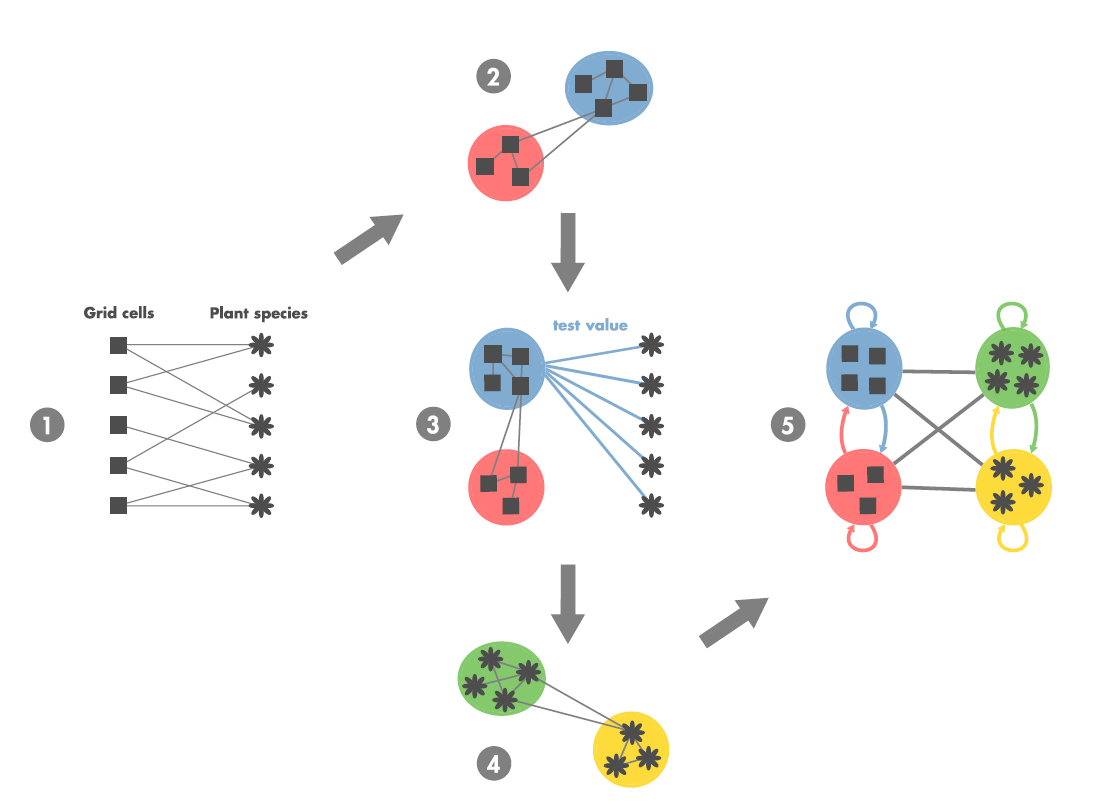
sp\_proj <- project\_network(sp\_mat, similarity = "simpson")  
sp\_proj <- sp\_proj[, c("id1", "id2", "simpson")]  
  
knitr::kable(head(sp\_proj))

|  |  |  |
| --- | --- | --- |
| id1 | id2 | simpson |
| 4 | 6 | 1.0000000 |
| 7 | 8 | 1.0000000 |
| 7 | 9 | 0.8571429 |
| 8 | 9 | 1.0000000 |
| 5 | 10 | 0.6666667 |
| 4 | 11 | 1.0000000 |

sp\_proj\_w <- project\_network(sp\_mat\_w, similarity = "bray")  
sp\_proj\_w <- sp\_proj\_w[, c("id1", "id2", "bray")]  
  
knitr::kable(head(sp\_proj\_w))

|  |  |  |  |
| --- | --- | --- | --- |
|  | id1 | id2 | bray |
| 2 | 2 | 1 | 0.0162964 |
| 3 | 3 | 1 | 0.0000000 |
| 4 | 4 | 1 | 0.0005101 |
| 5 | 5 | 1 | 0.0000020 |
| 6 | 6 | 1 | 0.0002640 |
| 7 | 7 | 1 | 0.0016733 |

knitr::include\_graphics("../figures/Lenormand\_et\_al\_2019\_Figure2.png")



# Community detection: Order Statistics Local Optimization Method

Running OSLOM. tp files containing the modularity results are directly stored within the virtual dataset, but the following chunk can be run independently.

Output of OSLOM are stored in a chosen directory and can be import into R with the command readRDS().

run\_oslom(sp\_proj, n\_runs = 5, t\_param = 0.1, cp\_param = 0.5,  
 saving\_directory = "D:/PIERRE\_DENELLE/CarHab/Bioregionalization\_extra/")  
  
res <- readRDS("D:/PIERRE\_DENELLE/CarHab/Bioregionalization\_extra/tp.rds")  
file.rename("D:/PIERRE\_DENELLE/CarHab/Bioregionalization\_extra/tp.rds",  
 "D:/PIERRE\_DENELLE/CarHab/Bioregionalization\_extra/tp\_binary.rds")  
# With weights  
run\_oslom(sp\_proj\_w, n\_runs = 5, t\_param = 0.1, cp\_param = 0.5,  
 saving\_directory = "D:/PIERRE\_DENELLE/CarHab/Bioregionalization\_extra/")  
  
res\_w <- readRDS("D:/PIERRE\_DENELLE/CarHab/Bioregionalization\_extra/tp.rds")

Converting the OSLOM .tp file into a list.

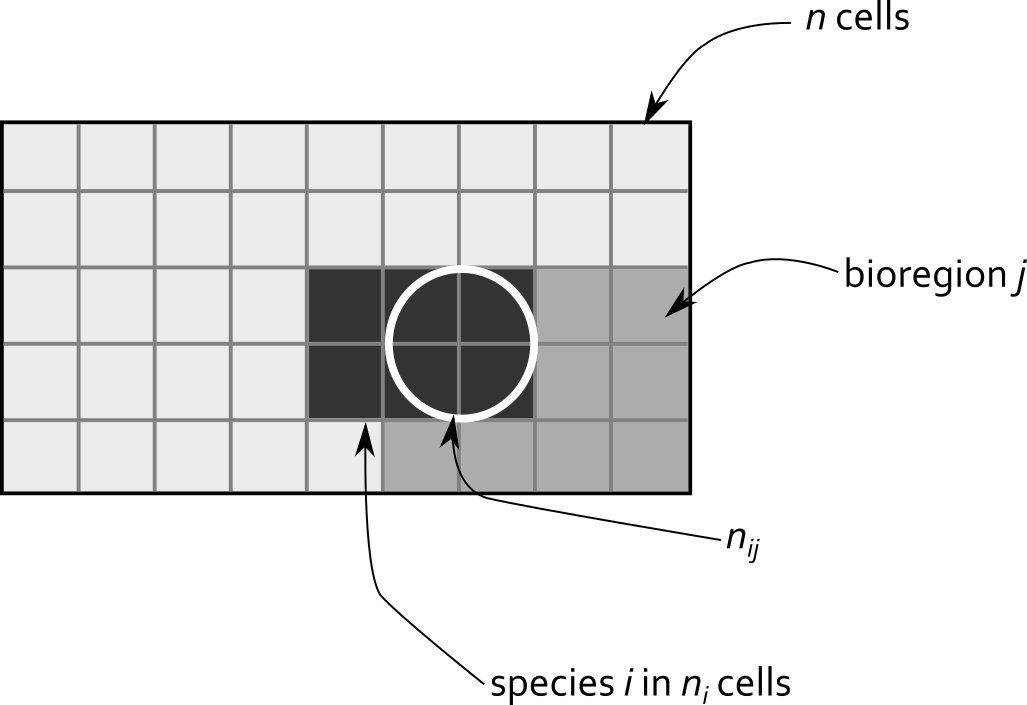
# Format OSLOM output into a data.frame  
oslom\_vignette <- oslom\_output(virtual[[2]], sp\_mat)  
# With weights  
oslom\_vignette\_w <- oslom\_output(virtual[[3]], sp\_mat\_w)  
  
print(paste0("Number of bioregions detected = ",  
 length(unique(oslom\_vignette$bioregion)),  
 "; and with weights: ",  
 length(unique(oslom\_vignette\_w$bioregion))))

## [1] "Number of bioregions detected = 9; and with weights: 6"

## Species’ contributions

Step 3 of Figure 1 (see Lenormand et al. (2019))

knitr::include\_graphics("../figures/zscore\_scheme.png")



The interactions between the different bioregions can then be calculated following these equations:

We normalize the rows:

Finally, we determine for each bioregion how each set of species contributes to it.

# 10 sites with species richness belonging to [1, 3]  
set.seed(1)  
sites <- sapply(paste0("site", seq(1:10)),  
 function(x) rep(x, sample(c(1:3), 1)))  
# Pool of 5 species distributed across the sites  
species <- lapply(sites,  
 function(x) sample(paste0("sp", seq(1:5)),   
 length(x), replace = TRUE))  
# 3 bioregions assigned randomly  
bioregions <- sample(paste0("bio", seq(1:3)), length(sites), replace = TRUE)  
bioregions <- data.frame(site = names(sites),  
 bioregion = bioregions)  
  
# Conversion to data frame  
fuzzy <- data.frame(site = as.character(unlist(sites)),  
 sp = as.character(unlist(species)))  
fuzzy <- left\_join(fuzzy, bioregions, by = "site")  
  
table(fuzzy$sp, fuzzy$bioregion)

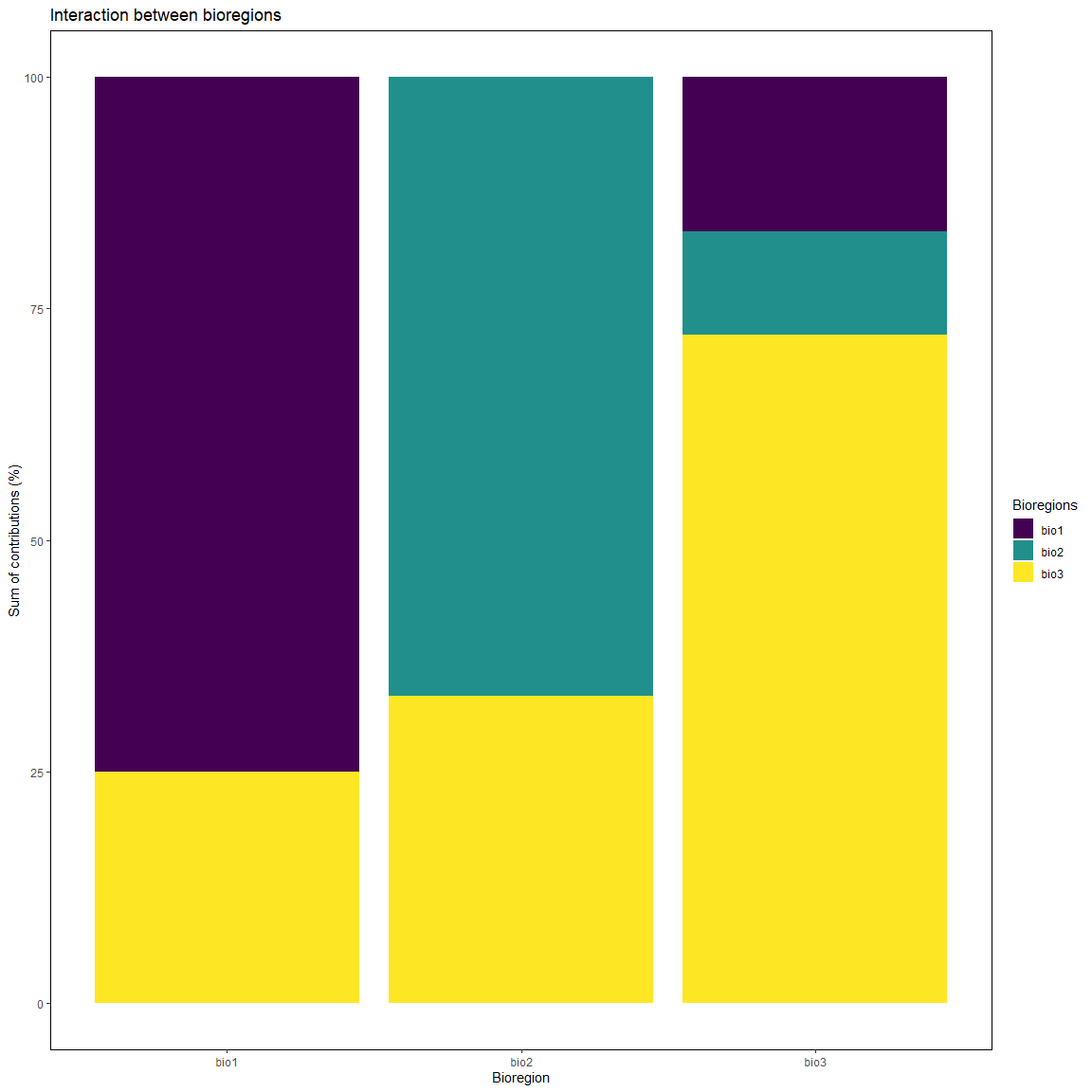
##   
## bio1 bio2 bio3  
## sp1 2 2 2  
## sp2 1 2 2  
## sp3 0 1 1  
## sp4 1 0 1  
## sp5 1 4 1

# Compute zscores  
zscores <- zscore(fuzzy, sp\_col = "sp", site\_col = "site",  
 bioregion\_col = "bioregion", output\_format = "matrix")  
zscores

## bioregion  
## sp bio1 bio2 bio3  
## sp1 0.2020305 -0.4082483 0.2020305  
## sp2 -0.5532833 0.0000000 0.5532833  
## sp3 -1.0497813 0.3535534 0.6998542  
## sp4 0.6998542 -1.4142136 0.6998542  
## sp5 -0.8081220 1.6329932 -0.8081220

# Interactions between bioregions  
interact(input\_network = "projected",  
 dat = zscores, plot = TRUE, output\_format = "matrix")

## $lambda  
## bio1 bio2 bio3  
## bio1 75.00000 0.00000 25.00000  
## bio2 0.00000 66.78141 33.21859  
## bio3 16.66667 11.18761 72.14572  
##   
## [[2]]



# tmp <- left\_join(sp\_df, oslom\_vignette, by = "site")  
tmp <- left\_join(sp\_df[which(sp\_df$pa > 0), ], oslom\_vignette, by = "site")  
# table(tmp$sp, tmp$bioregion)  
  
z\_scores <- zscore(tmp, sp\_col = "sp", site\_col = "site",  
 bioregion\_col = "bioregion", output\_format = "dataframe")  
  
top10 <- z\_scores %>%  
 group\_by(bioregion) %>%  
 top\_n(n = 10, zscore) %>% # extract top 10  
 mutate(rank = rank(-zscore, # ranking zcore in an ascending order  
 ties.method = "first")) %>% # if tie zscore, first species  
 dplyr::select(sp, bioregion, zscore, rank) %>%  
 mutate(zscore = round(zscore, 1)) %>% # rounding zscore to 1 digit  
 as.data.frame()  
  
knitr::kable(top10[which(top10$bioregion == "2"), ])

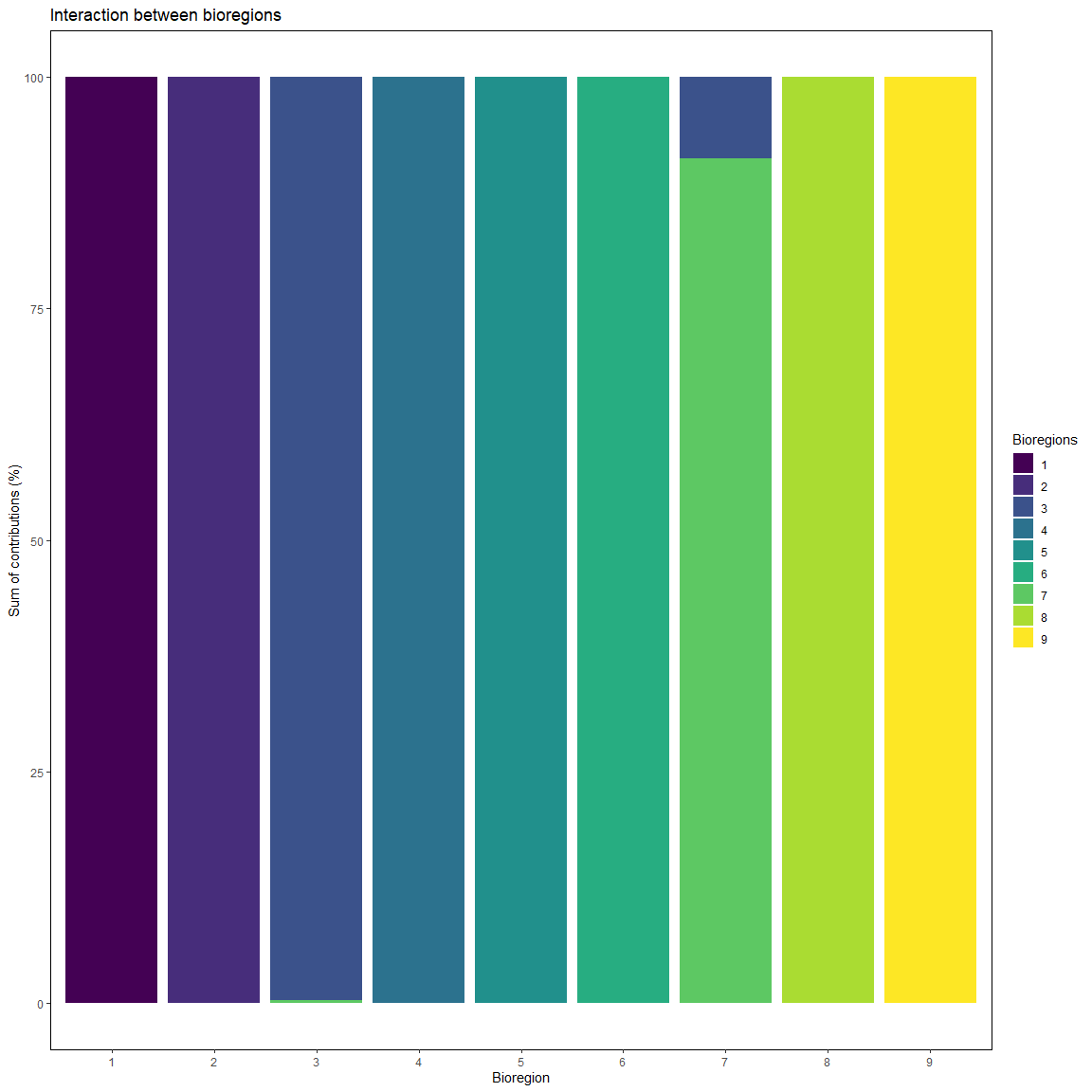
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | sp | bioregion | zscore | rank |
| 11 | sp51 | 2 | 13.5 | 10 |
| 12 | sp52 | 2 | 13.8 | 9 |
| 13 | sp53 | 2 | 15.2 | 8 |
| 14 | sp54 | 2 | 15.5 | 7 |
| 15 | sp55 | 2 | 16.3 | 6 |
| 16 | sp56 | 2 | 16.5 | 5 |
| 17 | sp57 | 2 | 17.1 | 4 |
| 18 | sp58 | 2 | 17.2 | 3 |
| 19 | sp59 | 2 | 18.8 | 1 |
| 20 | sp60 | 2 | 18.7 | 2 |

## Interaction between bioregions

z\_scores <- zscore(tmp, sp\_col = "sp", site\_col = "site",  
 bioregion\_col = "bioregion", output\_format = "matrix")  
  
lambda <- interact(input\_network = "projected",  
 dat = z\_scores, plot = TRUE, output\_format = "matrix")  
knitr::kable(head(lambda[[1]]))

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 100 | 0 | 0.00000 | 0 | 0 | 0 | 0.0000000 | 0 | 0 |
| 0 | 100 | 0.00000 | 0 | 0 | 0 | 0.0000000 | 0 | 0 |
| 0 | 0 | 99.72334 | 0 | 0 | 0 | 0.2766644 | 0 | 0 |
| 0 | 0 | 0.00000 | 100 | 0 | 0 | 0.0000000 | 0 | 0 |
| 0 | 0 | 0.00000 | 0 | 100 | 0 | 0.0000000 | 0 | 0 |
| 0 | 0 | 0.00000 | 0 | 0 | 100 | 0.0000000 | 0 | 0 |

# Plot  
lambda[[2]]



# Clustering

Example with Ward analysis and k-means clustering.

# CA\_res <- CA\_cluster(sp\_mat)  
ward\_res <- ward\_cluster(sp\_mat, K.max = 6)

# Community detection on bipartite networks

## Several algorithms

Bipartite algorithms applied on example dataset.

# With fastgreedy  
bip <- algo\_bipartite(dat = sp\_mat, algo = "greedy", weight = FALSE)  
  
# With Beckett algorithm  
bip2 <- algo\_bipartite(dat = sp\_mat, algo = "LPAwb", weight = FALSE)  
  
# With Infomap  
bip\_infomap <- algo\_bipartite(dat = sp\_mat, algo = "infomap", weight = FALSE)  
  
# Only sites  
bip\_site <- bip %>%  
 filter(cat == "site") %>%  
 rename(site = node) %>%  
 dplyr::select(site, module)  
  
bip\_site2 <- bip2 %>%  
 filter(cat == "site") %>%  
 rename(site = node) %>%  
 dplyr::select(site, module)  
  
bip\_site\_infomap <- bip\_infomap %>%  
 filter(cat == "site") %>%  
 rename(site = node) %>%  
 dplyr::select(site, module)

## Species and sites’ contributions

Cz computation on bipartite results.

bip\_cz <- bip[, c("node", "module", "cat")]  
colnames(bip\_cz) <- c("node", "mod", "cat")  
  
cz\_bip <- cz(dat = sp\_df[which(sp\_df$pa > 0), ], sp\_col = "sp",  
 site\_col = "site", bip = bip\_cz, ab = NULL)  
  
head(cz\_bip[[1]])

## node mod cat C n\_link\_mod mean\_link\_mod sd\_link\_mod z  
## 1 site101 1 site 0 6 12.59184 17.51985 -0.3762496  
## 2 site103 1 site 0 10 12.59184 17.51985 -0.1479372  
## 3 site108 1 site 0 10 12.59184 17.51985 -0.1479372  
## 4 site121 1 site 0 9 12.59184 17.51985 -0.2050153  
## 5 site123 1 site 0 6 12.59184 17.51985 -0.3762496  
## 6 site124 1 site 0 6 12.59184 17.51985 -0.3762496

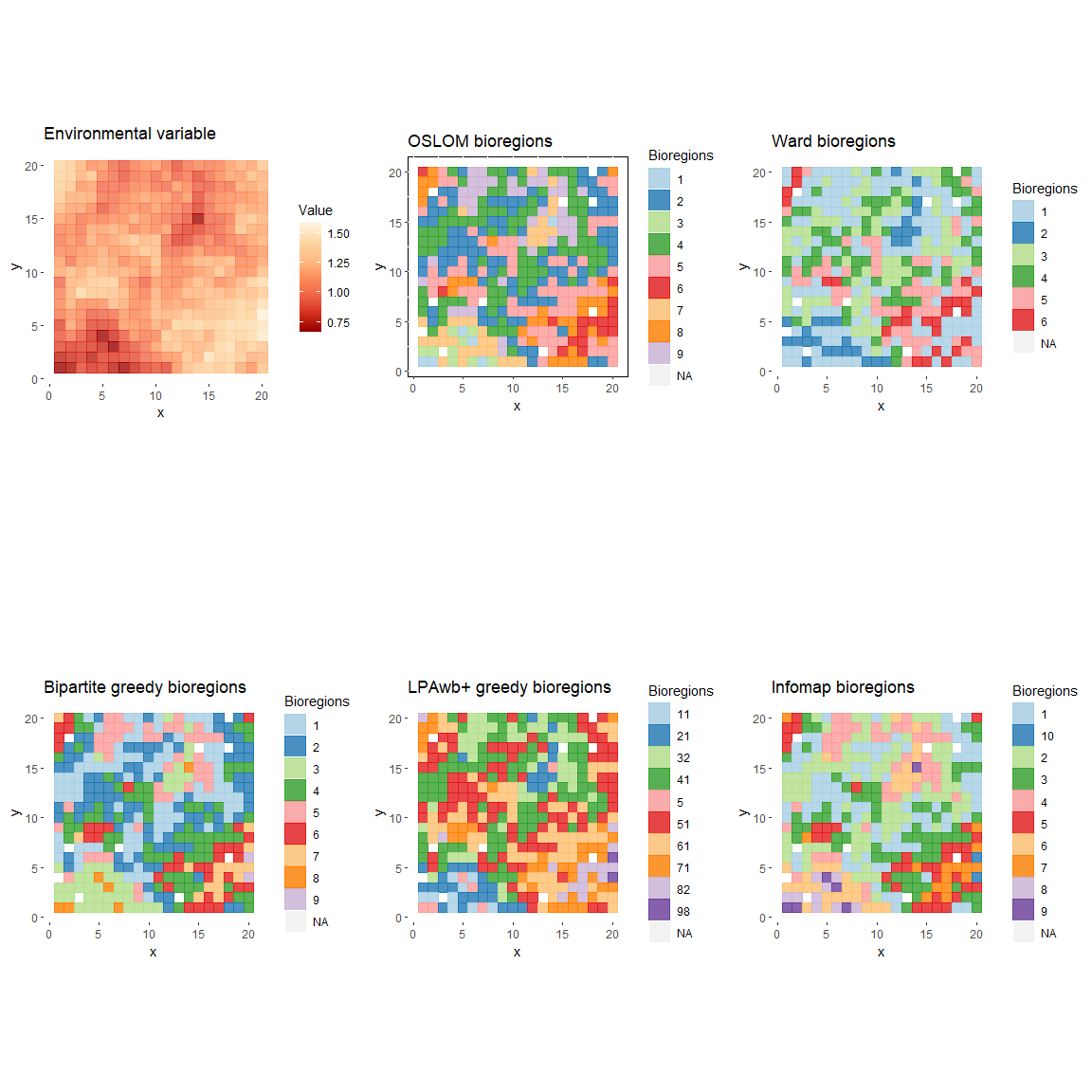
# Interaction  
# lambda <- interact(input\_network = "bipartite",  
# dat = link\_cz, plot = TRUE, output\_format = "matrix")  
# knitr::kable(head(lambda[[1]]))  
# # Plot  
# lambda[[2]]

# All contributions

tmp <- left\_join(sp\_df[which(sp\_df$pa > 0), ], oslom\_vignette, by = "site")  
# table(tmp$sp, tmp$bioregion)  
  
scores <- contribute(dat = tmp, sp\_col = "sp", site\_col = "site",  
 bioregion\_col = "bioregion")

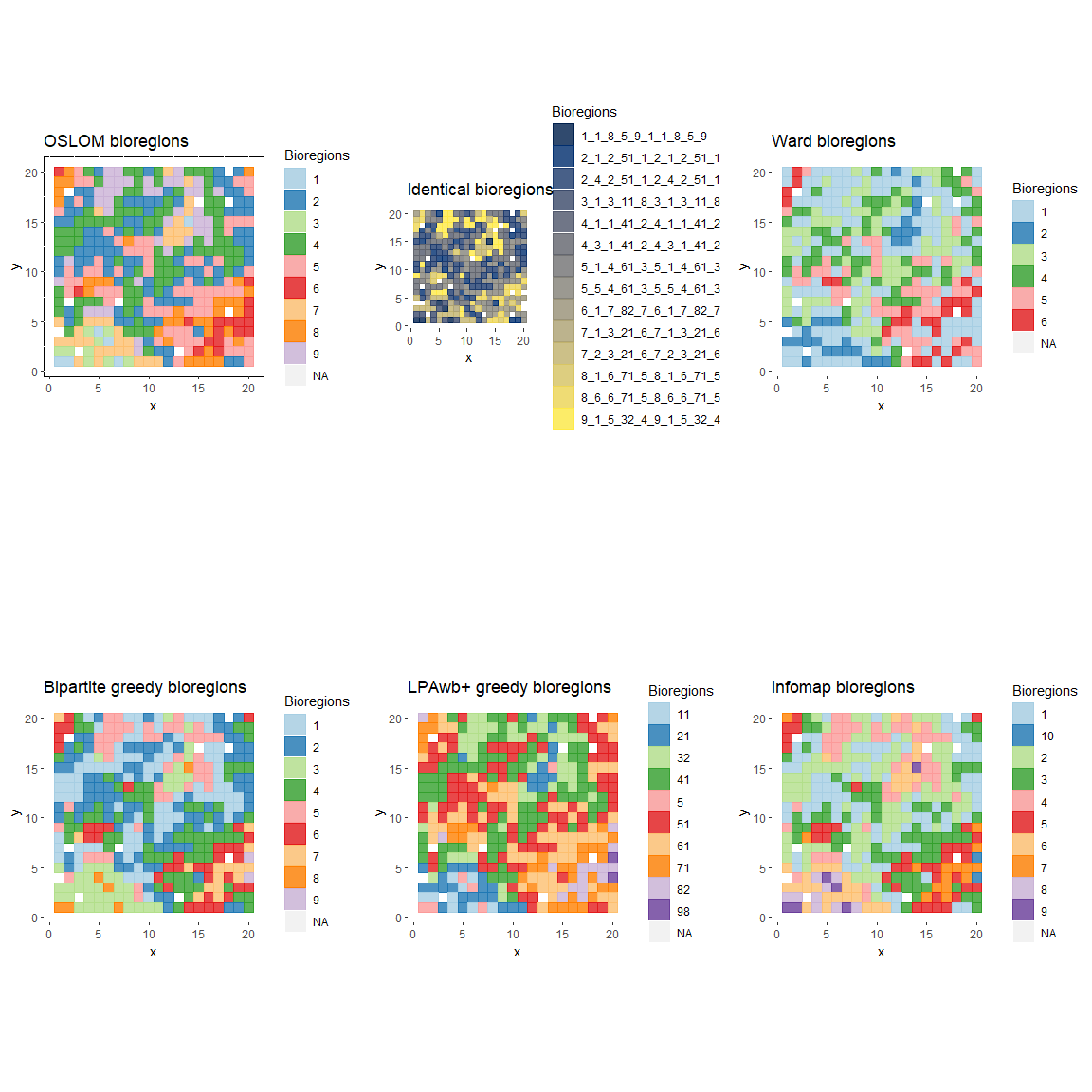
Projection on a map.

plot\_grid(  
 # Plot of environmental values  
 sp\_df %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = env, color = env),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_distiller("Value", palette = "OrRd") +  
 scale\_fill\_distiller("Value", palette = "OrRd") +  
 coord\_equal() +  
 labs(title = "Environmental variable") +  
 theme(panel.background = element\_rect(fill = "transparent",colour = NA)),  
   
 # Plot of OSLOM bioregions  
 sp\_df %>%  
 left\_join(oslom\_vignette, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(bioregion), color = as.factor(bioregion)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 # scale\_color\_viridis\_d("Bioregions", option = "E") +  
 # scale\_fill\_viridis\_d("Bioregions", option = "E") +  
 coord\_equal() +  
 labs(title = "OSLOM bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent",  
 colour = "black")),  
   
 # Plot of Ward bioregions  
 sp\_df %>%  
 left\_join(ward\_res, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(cluster), color = as.factor(cluster)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Ward bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of fastgreedy bioregions  
 sp\_df %>%  
 left\_join(bip\_site, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(module), color = as.factor(module)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Bipartite greedy bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of LPAwb bioregions  
 sp\_df %>%  
 left\_join(bip\_site2, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(module), color = as.factor(module)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "LPAwb+ greedy bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of Infomap bioregions  
 sp\_df %>%  
 left\_join(bip\_site\_infomap, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(module), color = as.factor(module)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Infomap bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 nrow = 2)

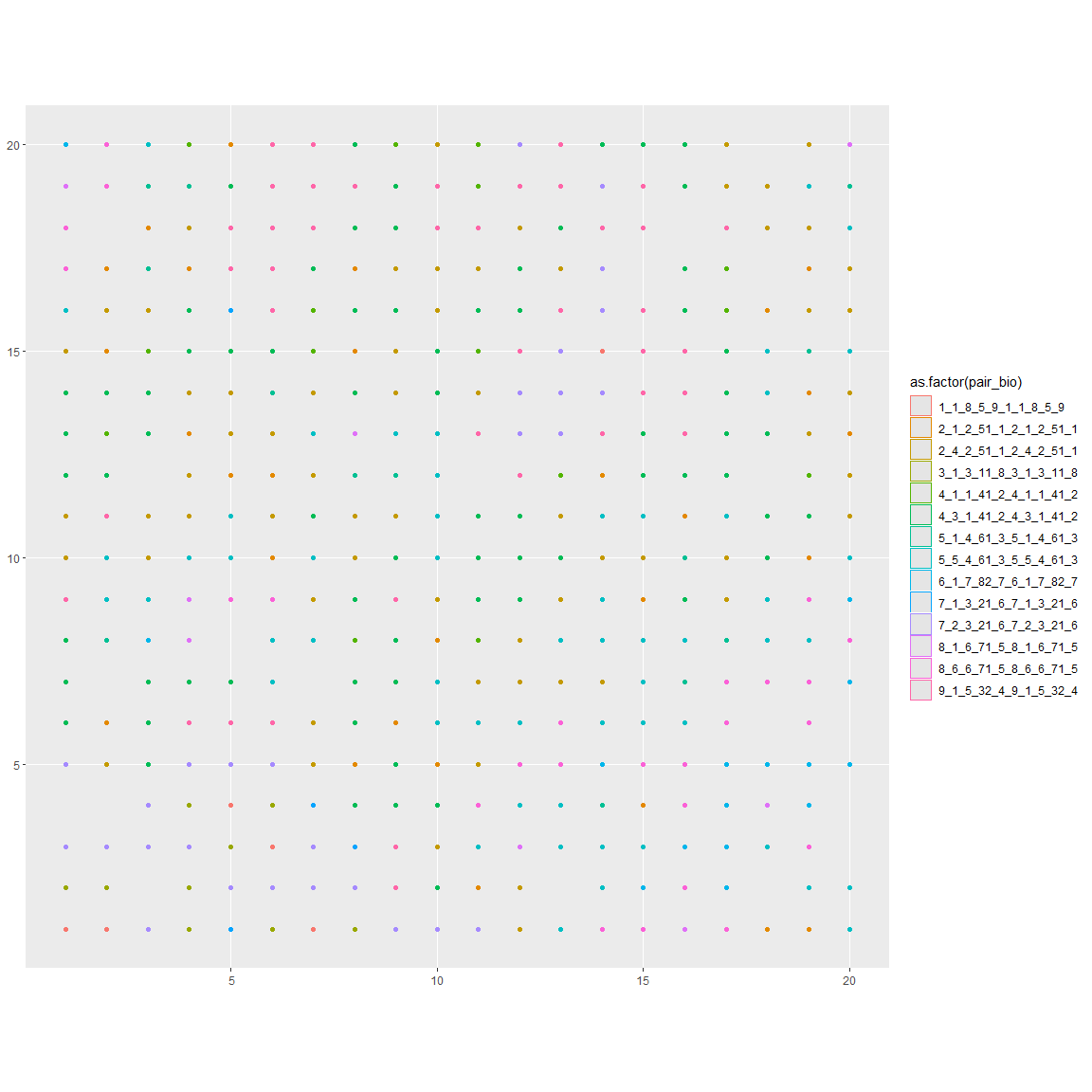


# Comparison of different bioregionalizations

# Gather all the bioregionalizations  
all\_bioregions <- sp\_df %>%  
 select(site, x, y, env) %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 left\_join(oslom\_vignette, by = "site") %>% # add OSLOM  
 rename(oslom = bioregion) %>%  
 left\_join(ward\_res, by = "site") %>% # add Ward  
 rename(ward = cluster) %>%  
 left\_join(bip\_site, by = "site") %>% # add greedy  
 distinct(site, .keep\_all = TRUE) %>%  
 rename(greedy = module) %>%  
 left\_join(bip\_site2, by = "site") %>% # add LPAwb  
 distinct(site, .keep\_all = TRUE) %>%  
 rename(lpawb = module) %>%  
 left\_join(bip\_site\_infomap, by = "site") %>% # add infomap  
 distinct(site, .keep\_all = TRUE) %>%  
 rename(infomap = module)  
  
# Test of comparison function  
all100 <- comparison(all\_bioregions, bio\_col = c(5:9))  
  
# Comparison of maps  
plot\_grid(  
 # Plot of OSLOM bioregions  
 sp\_df %>%  
 left\_join(oslom\_vignette, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(bioregion), color = as.factor(bioregion)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 # scale\_color\_viridis\_d("Bioregions", option = "E") +  
 # scale\_fill\_viridis\_d("Bioregions", option = "E") +  
 coord\_equal() +  
 labs(title = "OSLOM bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent",  
 colour = "black")),  
   
 # Identical pairs of plots  
 all100 %>%  
 left\_join(all\_bioregions[, c("site", "x", "y")], by = c("id1" = "site")) %>%  
 distinct(id1, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(pair\_bio), color = as.factor(pair\_bio)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_viridis\_d("Bioregions", option = "E") +  
 scale\_fill\_viridis\_d("Bioregions", option = "E") +  
 # scale\_color\_brewer("Bioregions", palette = "Paired") +  
 # scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Identical bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of Ward bioregions  
 sp\_df %>%  
 left\_join(ward\_res, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(cluster), color = as.factor(cluster)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Ward bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of fastgreedy bioregions  
 sp\_df %>%  
 left\_join(bip\_site, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(module), color = as.factor(module)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Bipartite greedy bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of LPAwb bioregions  
 sp\_df %>%  
 left\_join(bip\_site2, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(module), color = as.factor(module)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "LPAwb+ greedy bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 # Plot of Infomap bioregions  
 sp\_df %>%  
 left\_join(bip\_site\_infomap, by = "site") %>%  
 distinct(site, .keep\_all = TRUE) %>%  
 ggplot(aes(x, y)) +  
 geom\_tile(aes(fill = as.factor(module), color = as.factor(module)),  
 alpha = 0.8, width = 1, height = 1) +  
 scale\_color\_brewer("Bioregions", palette = "Paired") +  
 scale\_fill\_brewer("Bioregions", palette = "Paired") +  
 coord\_equal() +  
 labs(title = "Infomap bioregions") +  
 theme(panel.background = element\_rect(fill = "transparent", colour = NA)),  
   
 nrow = 2)



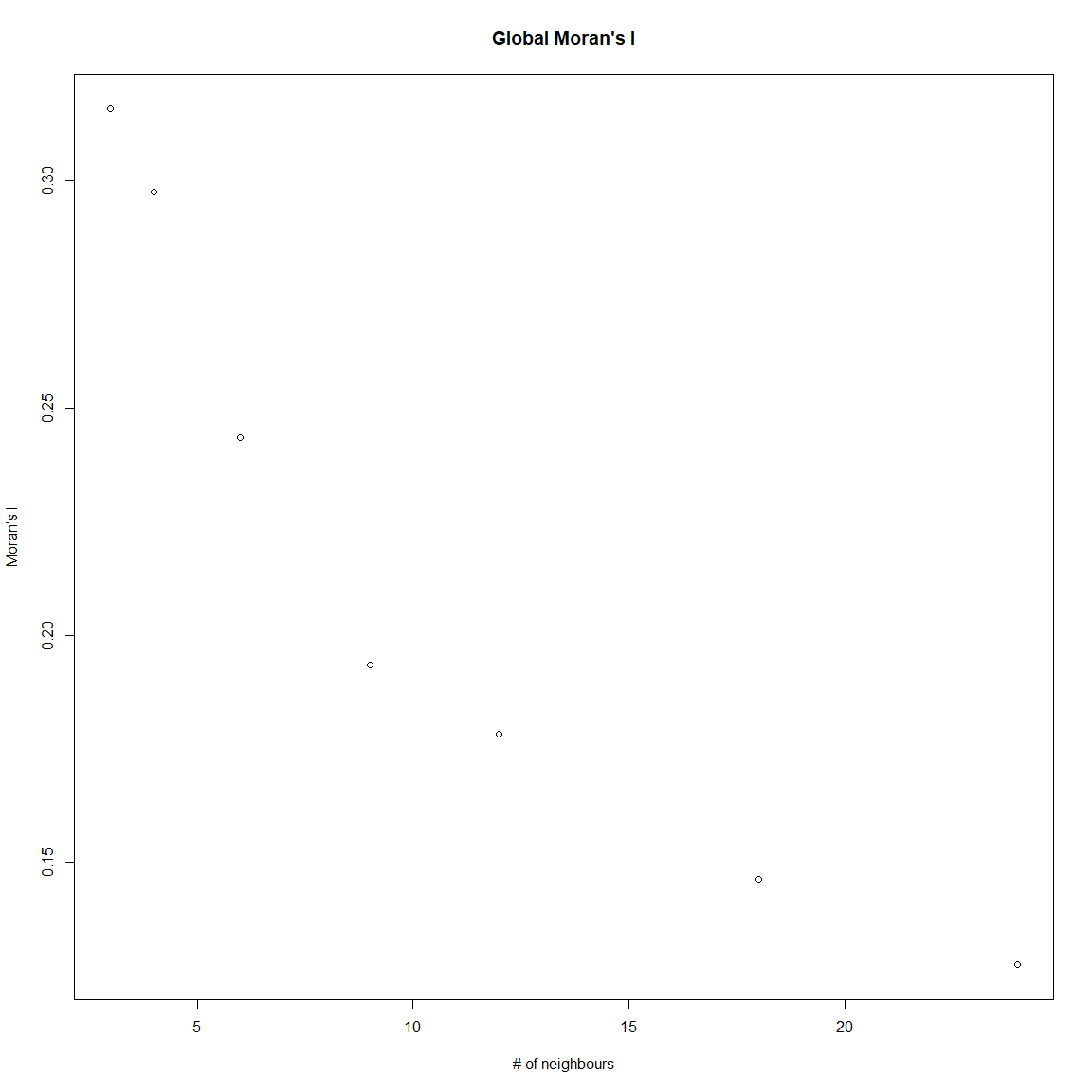
# Plot with tiles  
all100 %>%  
 left\_join(all\_bioregions[, c("site", "x", "y")], by = c("id1" = "site")) %>%  
 distinct(id1, .keep\_all = TRUE) %>%  
 st\_as\_sf(coords = c("x", "y")) %>%  
 group\_by(pair\_bio) %>%  
 st\_cast("MULTIPOINT") %>%  
 summarise() %>%  
 ggplot() +  
 geom\_sf(aes(color = as.factor(pair\_bio)))



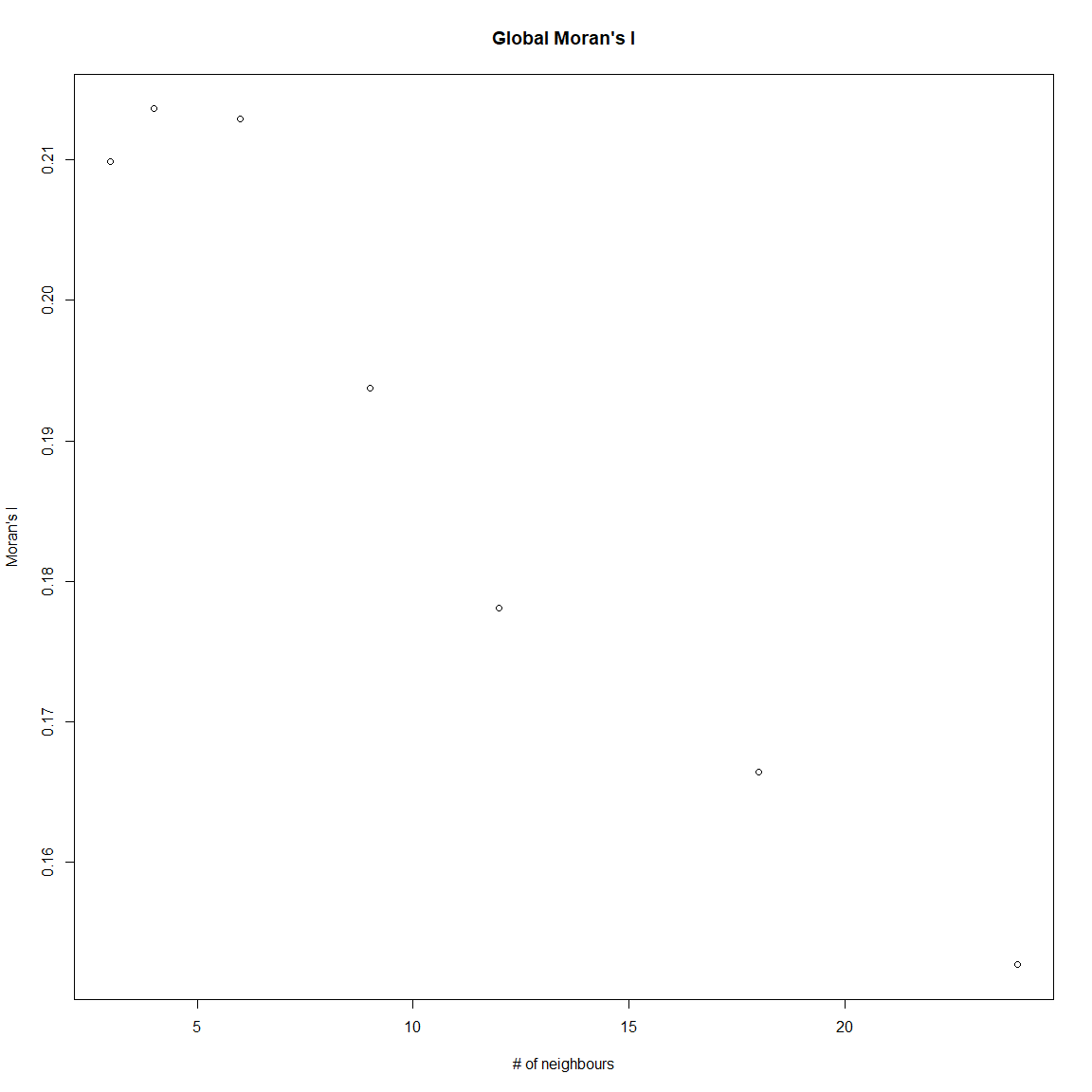
geom\_tile(aes(fill = as.factor(pair\_bio), color = as.factor(pair\_bio)),  
 alpha = 0.8, width = 1, height = 1)

## mapping: fill = ~as.factor(pair\_bio), colour = ~as.factor(pair\_bio)   
## geom\_tile: linejoin = mitre, na.rm = FALSE  
## stat\_identity: na.rm = FALSE  
## position\_identity

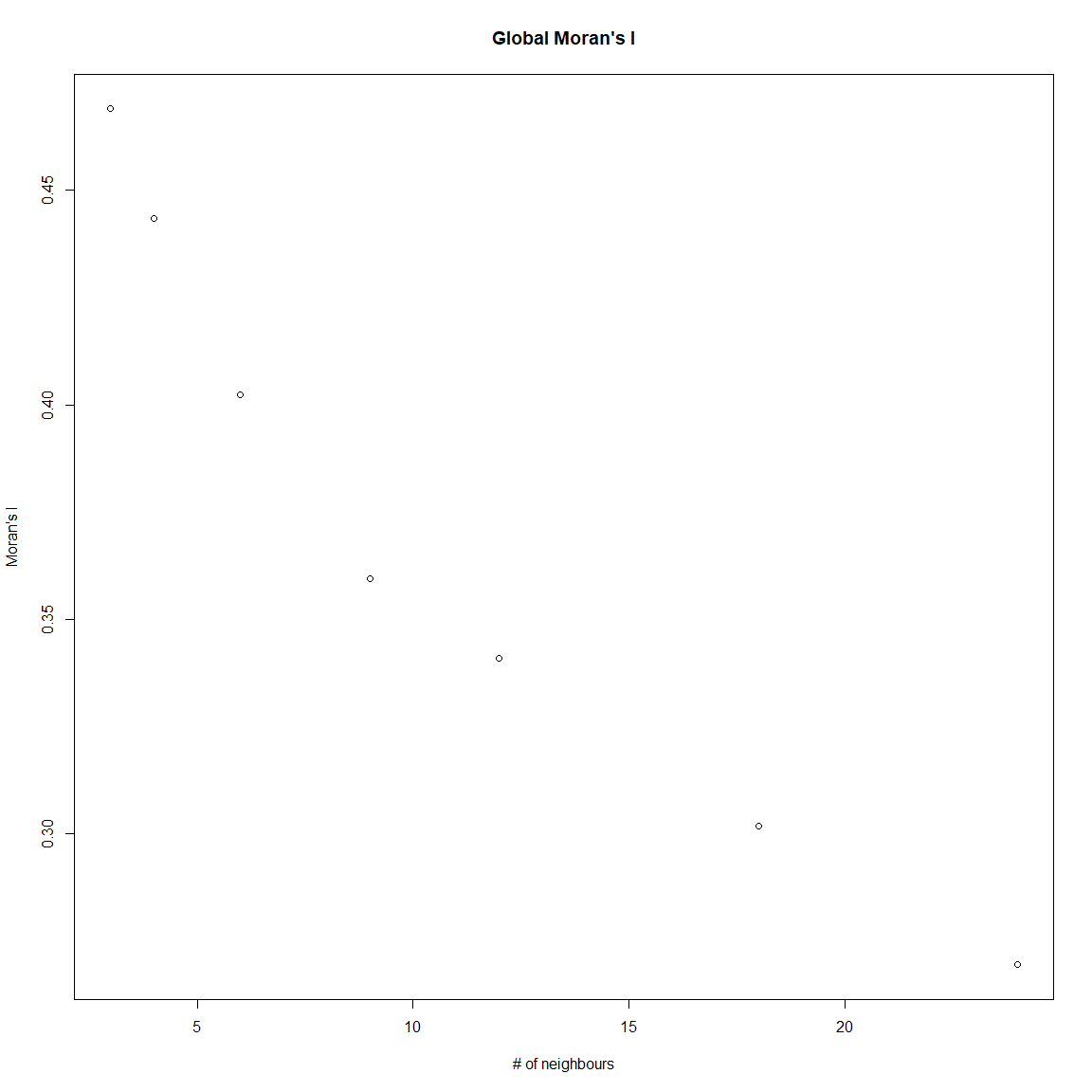
## Autocorrelation  
# https://mgimond.github.io/Spatial/spatial-autocorrelation-in-r.html  
library(sp)  
all\_bioregions\_sp <- all\_bioregions  
all\_bioregions\_sp <- all\_bioregions\_sp[complete.cases(all\_bioregions\_sp), ]  
coordinates(all\_bioregions\_sp) <- ~x+y  
  
all\_bioregions\_sp$oslom <- as.numeric(as.character(all\_bioregions\_sp$oslom))  
all\_bioregions\_sp$ward <- as.numeric(as.character(all\_bioregions\_sp$ward))  
all\_bioregions\_sp$greedy <- as.numeric(as.character(all\_bioregions\_sp$greedy))  
all\_bioregions\_sp$lpawb <- as.numeric(as.character(all\_bioregions\_sp$lpawb))  
all\_bioregions\_sp$infomap <- as.numeric(as.character(all\_bioregions\_sp$infomap))  
  
coo <- coordinates(all\_bioregions\_sp)  
# S.dist <- spdep::dnearneigh(coo, 0, 100)  
# lw <- spdep::nb2listw(S.dist, style="W",zero.policy=T)   
# MI <- spdep::moran.mc(all\_bioregions\_sp$oslom, lw, nsim=599,zero.policy=T)  
# plot(MI, main = "", las = 1)   
  
bws <- c(3, 4, 6, 9, 12, 18, 24)  
moran\_oslom <- lctools::moransI.v(coo, bws, all\_bioregions\_sp@data$oslom)



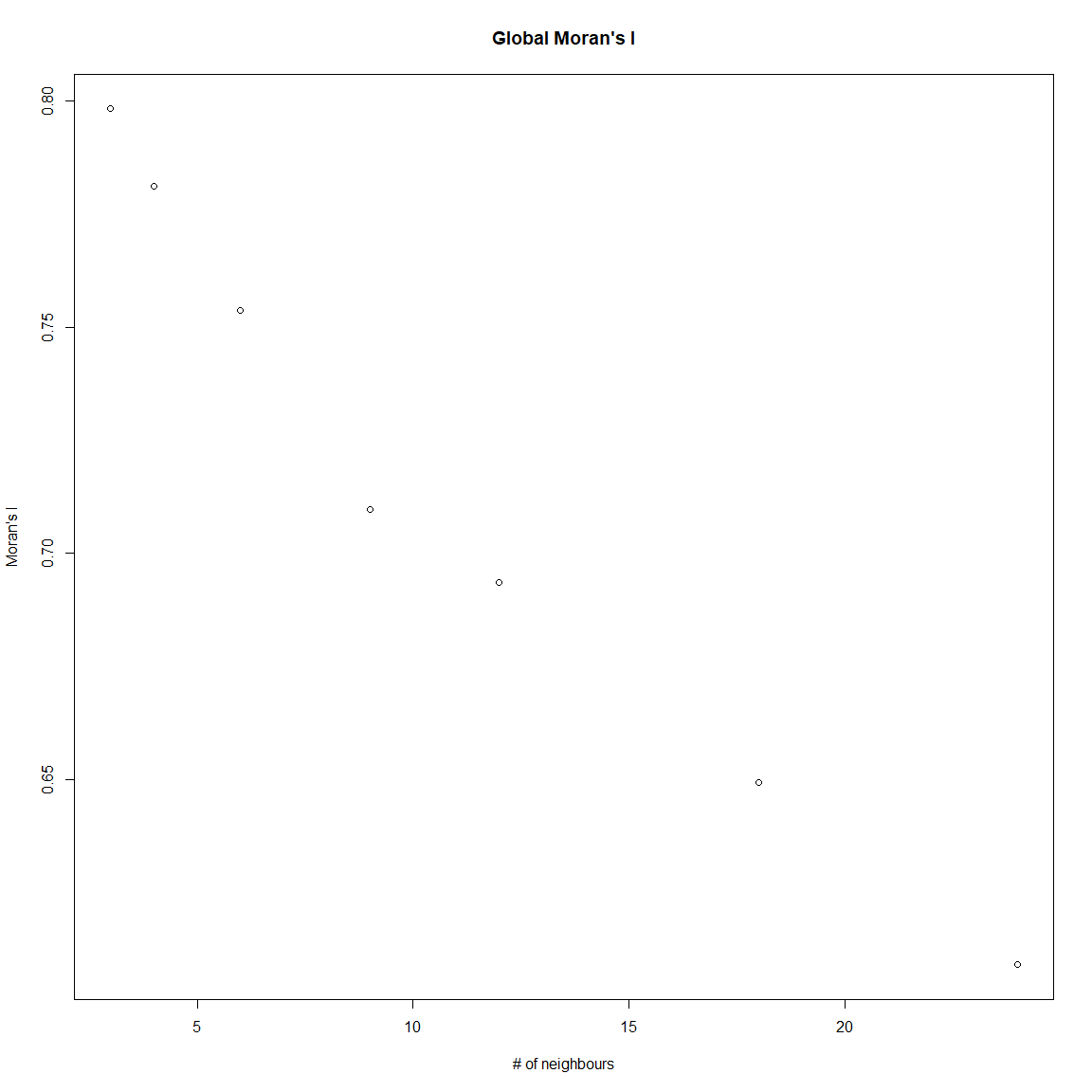
moran\_ward <- lctools::moransI.v(coo, bws, all\_bioregions\_sp@data$ward)



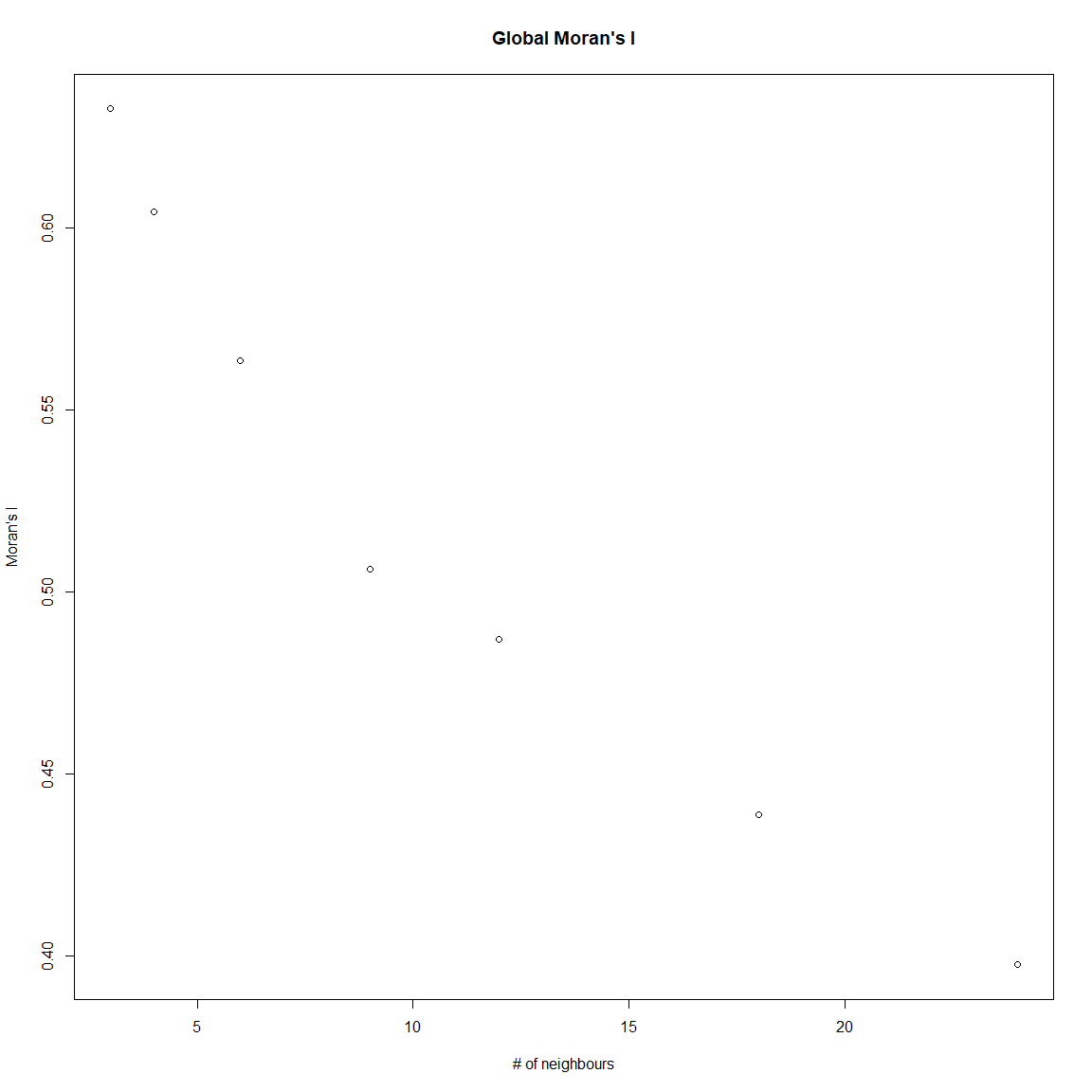
moran\_greedy <- lctools::moransI.v(coo, bws, all\_bioregions\_sp@data$greedy)



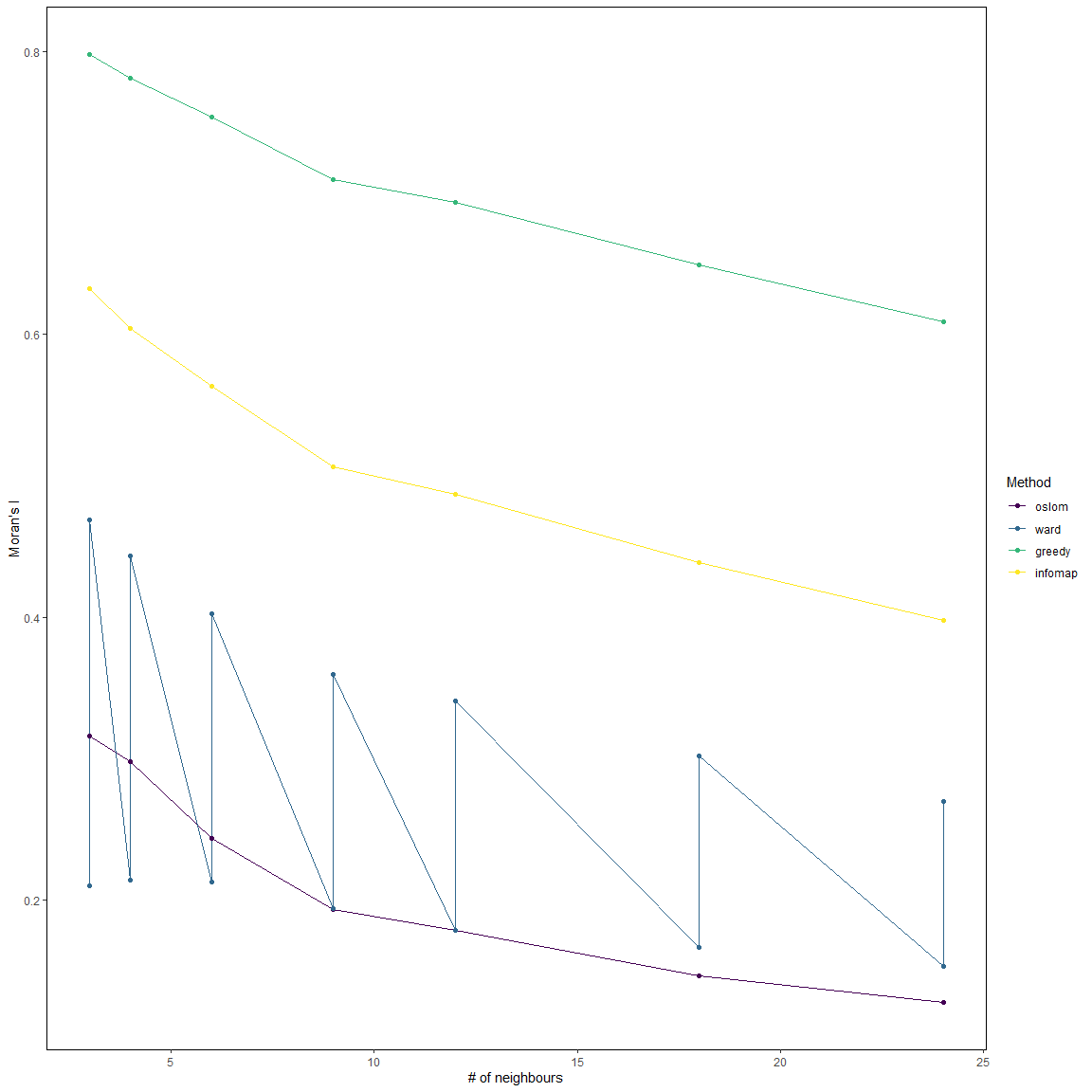
moran\_lpawb <- lctools::moransI.v(coo, bws, all\_bioregions\_sp@data$lpawb)



moran\_infomap <- lctools::moransI.v(coo, bws, all\_bioregions\_sp@data$infomap)



all\_moran <-  
 rbind(data.frame(  
 algo = "oslom", k = moran\_oslom[, "k"],  
 moran = moran\_oslom[, "Moran's I"]),  
 data.frame(algo = "ward", k = moran\_ward[, "k"],  
 moran = moran\_ward[, "Moran's I"]),  
 data.frame(algo = "ward", k = moran\_ward[, "k"],  
 moran = moran\_greedy[, "Moran's I"]),  
 data.frame(algo = "greedy", k = moran\_greedy[, "k"],  
 moran = moran\_lpawb[, "Moran's I"]),  
 data.frame(algo = "infomap", k = moran\_infomap[, "k"],  
 moran = moran\_infomap[, "Moran's I"])  
 )  
  
ggplot(all\_moran, aes(k, moran)) +  
 geom\_line(aes(color = as.factor(algo))) +  
 geom\_point(aes(color = as.factor(algo))) +  
 scale\_color\_viridis\_d("Method") +  
 labs(x = "# of neighbours", y = "Moran's I") +  
 theme\_classic() +  
 theme(panel.border = element\_rect(fill= NA, color = "black"))



We can as well compare each pair of bioregion to each other.

# sabre package: test between oslom and ward  
# Spatial Association Between REgionalizations  
library(sabre)  
library(sf)  
  
bioregions\_sf <- st\_as\_sf(all\_bioregions, coords = c("x", "y")) %>%  
 st\_cast(to = "MULTILINESTRING")  
  
po <- vmeasure\_calc(x = bioregions\_sf, x\_name = oslom,  
 y = bioregions\_sf, y\_name = ward)  
  
# Groups of pixels through the different methods  
list\_group\_oslom <- by(all\_bioregions$site, all\_bioregions$oslom, identity)  
list\_group\_oslom <- lapply(list\_group\_oslom, as.character)  
  
list\_group\_ward <- by(all\_bioregions$site, all\_bioregions$ward, identity)  
list\_group\_ward <- lapply(list\_group\_ward, as.character)  
  
list\_group\_greedy <- by(all\_bioregions$site, all\_bioregions$greedy, identity)  
list\_group\_greedy<- lapply(list\_group\_greedy, as.character)  
  
list\_group\_lpawb <- by(all\_bioregions$site, all\_bioregions$lpawb, identity)  
list\_group\_lpawb <- lapply(list\_group\_lpawb, as.character)  
  
list\_group\_infomap <- by(all\_bioregions$site, all\_bioregions$infomap, identity)  
list\_group\_infomap <- lapply(list\_group\_infomap, as.character)  
  
list\_combined <- lapply(list\_group\_oslom, function(x) x[x %in% list\_group\_ward[[1]]])  
list\_combined <- lapply(list\_combined, function(x) x[x %in% list\_group\_greedy[[1]]])  
list\_combined <- lapply(list\_combined, function(x) x[x %in% list\_group\_lpawb[[1]]])  
list\_combined <- lapply(list\_combined, function(x) x[x %in% list\_group\_infomap[[1]]])  
  
# For one combination only  
list1\_oslom <-  
 all\_bioregions[which(all\_bioregions$oslom == unique(all\_bioregions$oslom)[1]),  
 "site"]  
list1\_ward <- all\_bioregions[which(all\_bioregions$ward == unique(all\_bioregions$ward)[1]),  
 "site"]  
list1\_greedy <- all\_bioregions[which(all\_bioregions$greedy == unique(all\_bioregions$greedy)[1]),  
 "site"]  
list1\_lpawb <- all\_bioregions[which(all\_bioregions$lpawb == unique(all\_bioregions$lpawb)[1]),  
 "site"]  
list1\_infomap <- all\_bioregions[which(all\_bioregions$infomap == unique(all\_bioregions$infomap)[1]),  
 "site"]  
  
list1\_combined <- list1\_oslom[list1\_oslom %in% list1\_ward]  
list1\_combined <- list1\_combined[list1\_combined %in% list1\_greedy]  
list1\_combined <- list1\_combined[list1\_combined %in% list1\_lpawb]  
list1\_combined <- list1\_combined[list1\_combined %in% list1\_infomap]  
  
all\_bioregions$combined <- NA  
all\_bioregions[which(all\_bioregions$site %in% list1\_combined), "combined"] <- "1"

# Run all the steps together

All these functions can be called with the wrap-up all\_maps() function.

all\_res <- all\_maps(dat = sp\_df, form = "tidy", site, sp, ab = NULL,  
 binary = TRUE,  
 similarity = "simpson", network\_algo = "both",  
 saving\_directory,  
 bipartite\_algo = "greedy", weight = FALSE,  
 clustering = TRUE, ward\_method = "ward.D2",  
 optim\_method = "firstSEmax", nstart = 25, B = 50,  
 K.max = 20)