

62 Spatial autocorrelation scripts and calculations

63 *Written by Masahiro Ryo*

64 *2019 03 03*

65

---

## 66 Spatial analysis

67 Testing if any spatial autocorrelation structure appears in the distribution of each taxonomic  
68 group.

- 69 1. For each taxon, applying distance-based Moran's Eigenvector Mapping (dbMEM)
- 70 2. Opposingly, regressing each of the dbMEM patterns with the taxon data

### ## 1. Settings

Set the working directory appropriately.

```
r
setwd("C:/Users/masah/Dropbox/i_share/Research_share/2019_Danny_KHmicrobes/R/Danny_KH") getwd()
```

```
##                                                                 [1]
"C:/Users/masah/Dropbox/i_share/Research_share/2019_Danny_KHmicrobes/R/Danny_KH"
```

Calling some packages. Packages are automatically installed if needed.

```
r # install devtools tmp.install = require("devtools", character.only =
TRUE)==FALSE if(tmp.install) install.packages("devtools") # install patchwork
tmp.install = require("patchwork", character.only = TRUE)==FALSE if(tmp.install)
devtools::install_github("thomasp85/patchwork") # install and require them
package.list = c("ggplot2", "patchwork", "adespatial", "data.table", "mlr",
"parallelMap") tmp.install = which(lapply(package.list, require, character.only =
TRUE)==FALSE) if(length(tmp.install)>0)
install.packages(package.list[tmp.install], repos = "http://cran.us.r-
project.org") lapply(package.list, require, character.only = TRUE)
```

## 71 2. Data handling

72 Let's read the dataset:

```
73 df = fread("../Data/BIBS_DNA_kraken_Correct_Taxonomy_USEME.csv")
74
75
76 skip_data = 13
77 n_taxa = ncol(df) - skip_data
78 n_site = nrow(df)
```

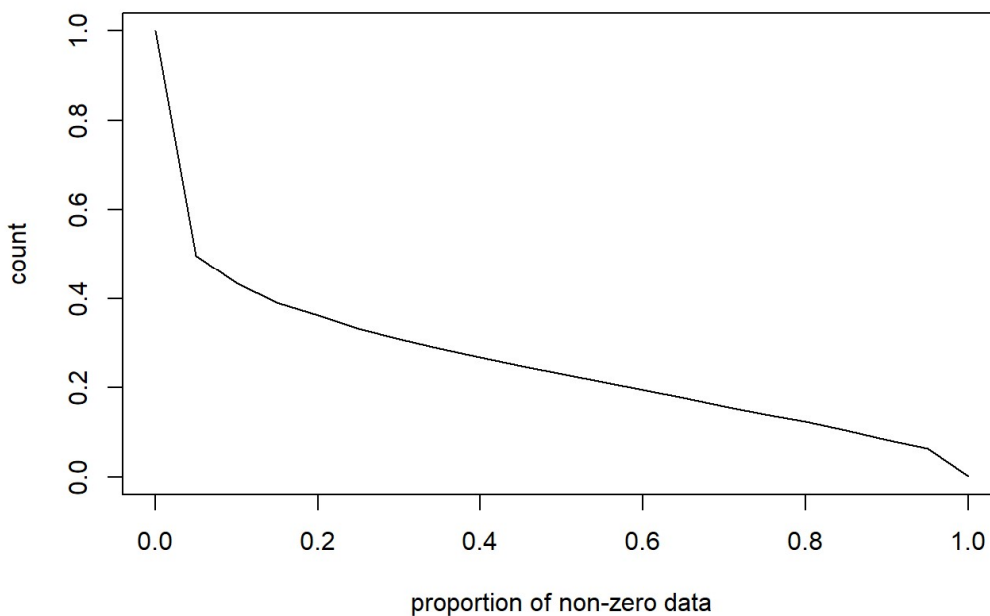
79 The number of taxa:  $1.352310^4$ .

80 The number of sites: 323

81

82 The data is mostly zero-inflated (many zero data). Visualizing the relationship between the  
83 proportion of zero-data and the number of taxa that exceeds the proportion.

```
84 cut_threshold = 0.9 # threshold for data occupation
85
86 n_count = numeric(0)
87 for(i in seq(0,1, by=0.05)){
88   n_count = append(n_count, sum(apply(df[, (skip_data+1):ncol(df)], 2,
89   function(x) sum(x>0)>=(i*nrow(df)))==TRUE))
90 }
91 plot(seq(0,1, by=0.05), n_count/n_taxa, type = "l", xlab="proportion of
92 non-zero data", ylab="count")
```



93

94 About the half of taxa (6761) are occupied by zero more than 95% of the sampling sites (n =  
95 323).

96 Here, we discard the taxa having zero-data more than 10% for further analyses.

```
97 id_selected = which(apply(df[, (skip_data+1):(skip_data+n_taxa)], 2,
98 function(x) sum(x>0)>=(cut_threshold*nrow(df))]==TRUE) + skip_data
99
100 df_selected = df[,..id_selected]
```

101 **The number of taxa is 1131 after the screening.**

102

---

### 103 **3. Spatial Autocorrelation Patterns across** 104 **multiple scales**

105 In this section, using [distance-based Moran's Eigenvector Mapping](#), we generate spatial  
106 autocorrelation patterns across multiple scales from the geocoordinates of the sampling sites.  
107 MEM vectors are screened according to its power to explain the spatial variability. See  
108 [Bauman D., Fortin M-J, Drouet T. and Dray S. \(2018\)](#)  
109 R package: [adespatial](#)

```
110 dbmem.data0 = dbmem(xyORdist=cbind(df$Lon,df$Lat))
111 dbmem.selected0 = mem.select(x=cbind(df$Lon,df$Lat), listw =
112 attributes(dbmem.data0)$listw)
113 ## Procedure stopped (R2more criteria): variable 6 explains only 0.000620
114 of the variance.
115 print(dbmem.selected0$summary)
116 ##   variables order      R2      R2Cum AdjR2Cum pvalue
117 ## 1     MEM1      1 0.593704130 0.5937041 0.5924384 0.001
118 ## 2     MEM3      3 0.207731438 0.8014356 0.8001945 0.001
119 ## 3     MEM2      2 0.134851233 0.9362868 0.9356876 0.001
120 ## 4     MEM4      4 0.009006865 0.9452937 0.9446055 0.001
121 ## 5     MEM5      5 0.002362262 0.9476559 0.9468303 0.002
122 dbmem.selected =
123 dbmem.selected0$summary$variables[which(dbmem.selected0$summary$R2>0.10)]
```

124

125 In total, 6 MEM vectors were produced.  
126 Yet, we focus only on positive autocorrelation patterns of which R2 is higher than 10%.  
127 We use MEM1, MEM3, MEM2 for further analyses.

```
128 dbmem.data = data.frame(cbind(df$Lon, df$Lat,
129 dbmem.data0[,dbmem.selected]))
130 colnames(dbmem.data) = c("Longitude", "Latitude",
131 colnames(dbmem.data0[,dbmem.selected]))
132
133 df = cbind(df, dbmem.data)
```

134

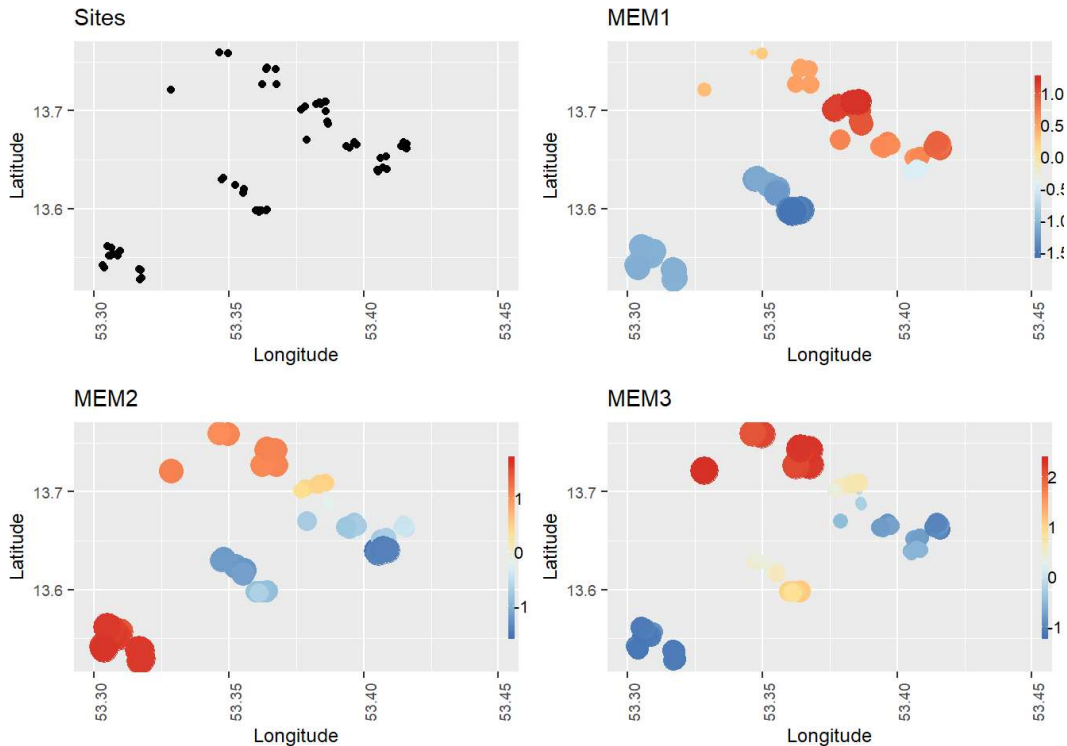
135 We plot the geocoordinates of the sampling sites and the selected MEM vectors.

```
136 # plot MEMs
137
138 g_0 = ggplot(data=dbmem.data, aes(x = Longitude, y=Latitude)) +
139   geom_point()+
140   xlim(c(53.3,53.45)) +
141   ggtitle('Sites') +
142   theme(text = element_text(size=9),
143         axis.text.x = element_text(angle=90, hjust=1))
144
145 g = ggplot(data=dbmem.data, aes(x = Longitude, y=Latitude)) +
146   scale_color_distiller(palette = "RdYlBu") +
147   guides(size=FALSE) +
148   xlim(c(53.3,53.45)) +
149   theme(
150     legend.title=element_blank(),
151     legend.text = element_text(size=8),
152     legend.key.width = unit(3, "point"),
153     text = element_text(size=9),
154     axis.text.x = element_text(angle=90, hjust=1),
```

```

155     legend.justification = c(0.5, 0.5),
156     legend.position = c(1, 0.5),
157     legend.background = element_rect(fill=alpha('white', 0.0))
158   )
159   for(i in 1:length(dbmem.selected)){
160     eval(parse(text=(paste(
161       "g_",i, "= g + geom_point(aes(size = abs(MEM",i,"), color = MEM",i,"))
162 + ggtitle('MEM", i,"')",
163       sep=""))))
164   }
165   print((g_0|g_1) / (g_2 | g_3))

```



166

167

## 168 4. Regressing each SAC patterns (MEM 169 vectors) using the taxa data

170 To explore which taxa reveal a strong association with each SAC pattern, we use a random  
171 forest algorithm. Since a relatively large instability is expected due to the excessive amount  
172 of predictors compared to the sample size, we use 10-fold subsampling to assess the relative  
173 importance of the predictors and model accuracy.

```

174   for(j in dbmem.selected){ #length(dbmem.selected)
175     # selecting a MEM vector and combine it with the selected taxa data
176     #j = dbmem.selected[1]
177     df.2 = data.frame(dbmem.data0[,j],df_selected)
178     colnames(df.2) = c(j, colnames(df_selected))
179   }

```

```

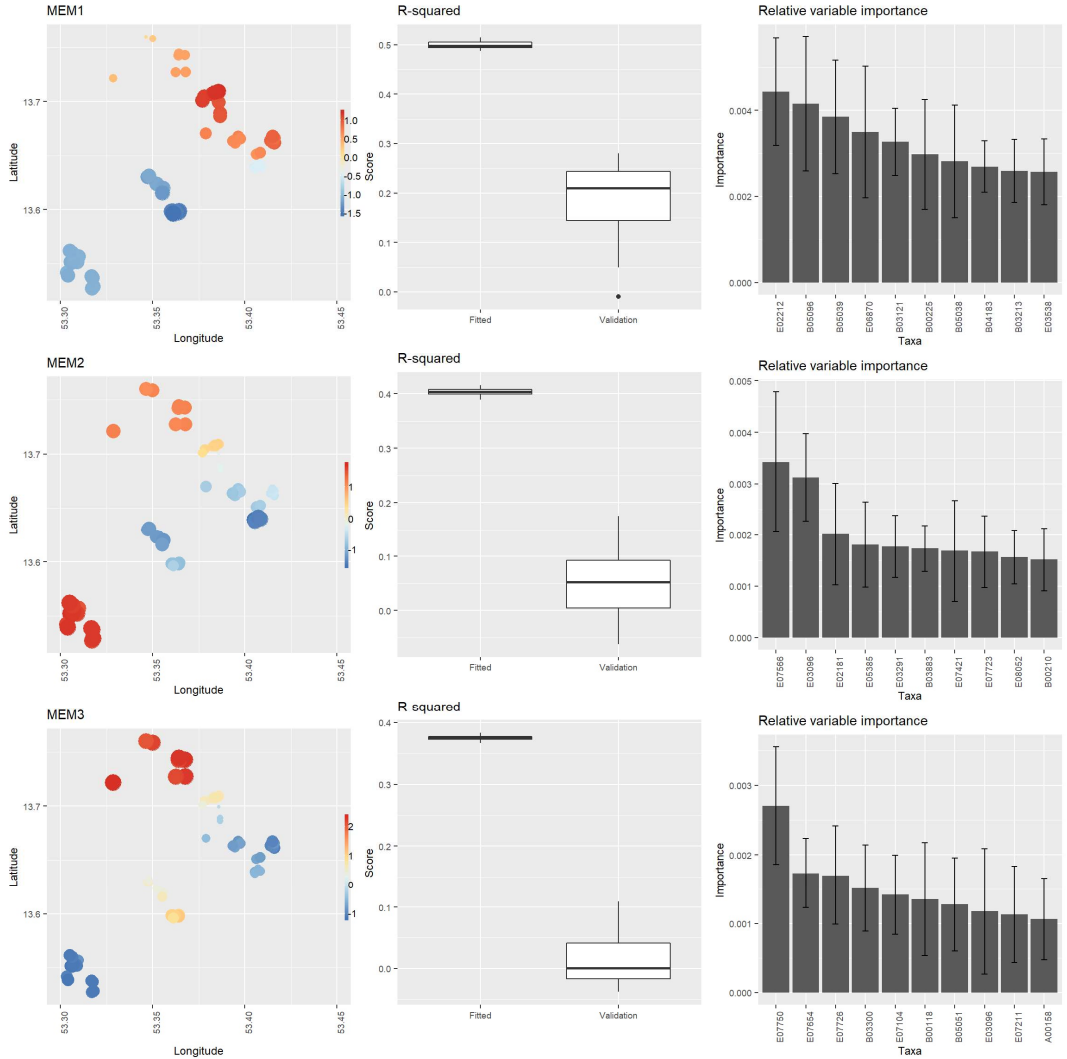
180     # iteration group
181     k_fold = 10
182     set.seed(2)
183     i_row = sample(c(1:k_fold), nrow(df.2), replace=T)
184     R2.fitting = numeric(0)
185     R2.validation = numeric(0)
186     vimp = matrix(NA, ncol = k_fold, nrow = length(df_selected))
187     rownames(vimp) = colnames(df_selected)
188
189     # parallelization
190     parallelStartSocket(3)
191
192     # RF modeling
193     for(i_group in c(1:k_fold)){
194         regr.task = makeRegrTask(data = df.2, target = j)
195         regr.learner.cforest = makeLearner(cl="regr.cforest", predict.type
196 = "response", ntree = 1000)
197         model.cforest = train(regr.learner.cforest, regr.task, subset =
198 which(i_row!=i_group))
199         fit.cforest = predict(model.cforest, task = regr.task, subset =
200 which(i_row!=i_group))
201         pred.cforest = predict(model.cforest, task = regr.task, subset =
202 which(i_row==i_group))
203         R2.fitting = append(R2.fitting, performance(fit.cforest,
204 measures = list(rsq)))
205         R2.validation = append(R2.validation, performance(pred.cforest,
206 measures = list(rsq)))
207         vimp[,i_group] = generateFilterValuesData(regr.task, method =
208 "cforest.importance")$data$cforest.importance
209     }
210     parallelStop()
211     # calculating the mean and standard deviation for each taxon
212     vimp_summary = data.frame(id = colnames(df_selected), ave =
213 apply(vimp,1,mean), stdev = apply(vimp,1,sd))
214     vimp_summary = vimp_summary[order(vimp_summary$ave, decreasing = T),]
215
216     # storing the R2 scores in the form of ggplot
217     g = ggplot(data.frame(eval=append(rep("Fitted",
218 k_fold),rep("Validation", k_fold)),
219 score=append(R2.fitting, R2.validation)),
220 aes(x=eval, y=score)) +
221     geom_boxplot() +
222     labs(title="R-squared", x="", y="Score") +
223     theme(text = element_text(size=9))
224     eval(parse(text=(paste("g_eval_",j, "= g", sep="")))))
225
226     # storing variable importance measure in the form of ggplot
227     g = ggplot(vimp_summary[1:10, ], aes(x=reorder(id, -ave), y=ave)) +
228     geom_bar(stat="identity") +
229     geom_errorbar(aes(ymin=ave-stdev, ymax=ave+stdev), width=.2,
230 position=position_dodge(.9)) +
231     labs(title="Relative variable importance", x="Taxa",
232 y="Importance") +
233     theme(text = element_text(size=9), axis.text.x =
234 element_text(angle=90, hjust=1))
235
236     eval(parse(text=(paste("g_vimp_",j, "= g", sep="")))))
237
238 }
239 ## Starting parallelization in mode=socket with cpus=3.
240 ## Stopped parallelization. All cleaned up.

```

```

241 ## Starting parallelization in mode=socket with cpus=3.
242 ## Stopped parallelization. All cleaned up.
243 ## Starting parallelization in mode=socket with cpus=3.
244 ## Stopped parallelization. All cleaned up.
245 print(g_1 / g_2 / g_3 | g_eval_MEM1 / g_eval_MEM2/ g_eval_MEM3 |
246 g_vimp_MEM1 / g_vimp_MEM2 / g_vimp_MEM3)

```



247

## 248 Interpretation

249 None of the analyzed predictors revealed a strong match to any SAC patterns.  
 250 Supporting this, cross Validation showed a strong decrease of R2 to nearly zero.  
 251 Probably, no taxa have a strong spatial pattern in this region.

252

253

254