***l. Индекс пространственной автокорреляции Морана***

library(readxl)

data\_2015 <- read\_excel("DATA.xlsx", sheet = "2015")

data\_2016 <- read\_excel("DATA.xlsx", sheet = "2016")

data\_2017 <- read\_excel("DATA.xlsx", sheet = "2017")

data\_2018 <- read\_excel("DATA.xlsx", sheet = "2018")

data\_2019 <- read\_excel("DATA.xlsx", sheet = "2019")

head(data\_2019)

library(ape)

data\_2019.dists <- as.matrix(dist(cbind(data\_2019$lat, data\_2019$lon)))

data\_2019.dist.inv <- 1/data\_2019.dists

diag(data\_2019.dist.inv) <- 0

data\_2019.dist.inv[1:5, 1:5]

Moran.I(data\_2019$MigrPr\_19, data\_2019.dist.inv)

Moran.I(data\_2019$ZP\_19, data\_2019.dist.inv)

Moran.I(data\_2019$VvodDom\_19, data\_2019.dist.inv)

Moran.I(data\_2019$VvodDom\_19, data\_2019.dist.inv,na.rm=TRUE)

# Оценка индекса Могана на основе двоичной матрицы расстояний

data\_2019.dist.bin <- (data\_2019.dists > 0 & data\_2019.dists <= .75)

Moran.I(data\_2019$ZP\_19, data\_2019.dist.bin)

**ll. Построение авторегрессионной пространственной модели**

# Загрузка данных

library(readxl)

data\_2015 <- read\_excel("DATA.xlsx", sheet = "2015")

data\_2016 <- read\_excel("DATA.xlsx", sheet = "2016")

data\_2017 <- read\_excel("DATA.xlsx", sheet = "2017")

data\_2018 <- read\_excel("DATA.xlsx", sheet = "2018")

data\_2019 <- read\_excel("DATA.xlsx", sheet = "2019")

library(gstat)

library(sp)

data <- as.data.frame(data\_2019)

coordinates(data) <- ~lon+lat

plot(data)

hscat(VvodOF\_19 ~ 1, data, (0:9) \* 10)

hscat(log(VvodOF\_19) ~ 1, data, (0:9) \* 10)

library(spdep)

library(spatialreg)

library(adespatial)

xy <- as.data.frame(coordinates(data))

names(xy) <- c('x', 'y')

# Поиск границ на основе всех моделей

candidates <- listw.candidates(coord = xy)

for(x in names(candidates))

{

print('')

print(x)

print(summary(candidates[[x]]))

}

names(candidates)

plot(candidates$Gabriel\_Binary, xy)

# Поиск границ на основе модели "gab" и "bin" модели весов

candidates <- listw.candidates(coord = xy, nb = "gab", weights = "bin")

# Выбор первой (и единственной модели)

lw <- candidates[[1]]

plot(lw, xy)

# Расчёт Spatial simultaneous autoregressive model estimation by maximum likelihood

res <- lagsarlm(VvodOF\_19 ~ Inv\_19 + ZP\_19, data=data, listw = lw)

# Суммарная информация

summary(res)

# Суммарная информация с корреляцией

summary(res, correlation=TRUE)

# Расчёт Bayesian MCMC spatial simultaneous autoregressive model estimation

res.lag <- spBreg\_lag(VvodOF\_19 ~ Inv\_19 + ZP\_19, data=data, listw = lw)

summary(res.lag)

plot(res.lag)

##########################

# Картограммы РФ

library(ggplot2)

library(maptools)

library(dplyr)

library(sf)

# Загрузка файла ru формата ESRI

map <- st\_read("ru.shp")

# Карты зачастую сделаны неверно, есть пересечение субъектов, а такого быть не должно

# Исправим эту проблему

# Подключим пакет lwgeom

library(lwgeom)

# Сделаем карту правильной (без пересечений)

map <- st\_make\_valid(map)

# Какие переменные есть в данных этой карты

names(map)

# Столбик id из данных этой карты

map$id

# Загрузка ИПЦ по субъектам РФ

cp <- read\_excel('ru2.xlsx')

names(cp)

# Сделаем из числа, строку, для того чтобы использовать для ручного заполнения

cp$rank <- as.character(cp$Rank)

# Соединение карты и ИПЦ на основе столбика id в переменную map\_cp

map %>% merge(cp, by='id') -> map\_cp

# Еще раз названия переменных, но уже из map\_cp

names(map\_cp)

# Посмотрим на содержание переменной map\_cp

map\_cp

# 1 - зеленый

# 2 - голубой

# 3 - желтый

# 4 - красный

# Увеличим размер легенды по горизонтали (...width) и вертикали (...size)

ggplot(map\_cp) +

geom\_sf(aes(fill=rank)) +

theme\_void() +

scale\_fill\_manual(values=c("1"="palegreen2", "2"="dodgerblue1", "3"="gold", "4"="coral", "0"="gray" ))

***lll. Пространственно-динамическое моделирование***

# Загрузка данных

library(readxl)

# Загрузка данных в каждую отдельную таблицу

data\_2015 <- read\_excel("DATA.xlsx", sheet = "2015")

data\_2016 <- read\_excel("DATA.xlsx", sheet = "2016")

data\_2017 <- read\_excel("DATA.xlsx", sheet = "2017")

data\_2018 <- read\_excel("DATA.xlsx", sheet = "2018")

data\_2019 <- read\_excel("DATA.xlsx", sheet = "2019")

library(dplyr)

# Объединение в единый массив данных, при этом порядок сортировки для строк на основе наименования города

# для того чтобы не добавлялись дублирующие поля "lon", "lat", "city" у всех наборов данных, кроме первого

# убираются эти поля

data <- cbind(

data\_2015 %>% arrange(city) ,

data\_2016 %>% arrange(city) %>% select(-lon,-lat,-city),

data\_2017 %>% arrange(city) %>% select(-lon,-lat,-city),

data\_2018 %>% arrange(city) %>% select(-lon,-lat,-city),

data\_2019 %>% arrange(city) %>% select(-lon,-lat,-city)

)

# Задание проекции WGS84 Russia

crsRU <- CRS("+proj=laea +lat\_0=90 +lon\_0=90 +x\_0=0 +y\_0=0 +datum=WGS84 +units=m +no\_defs")

# Задание проекции WGS84

crs84 <- CRS("+proj=longlat +datum=WGS84")

#### Загрузка данных из отдельных векторов значений

sp = cbind(x = data$lon, y = data$lat)

row.names(sp) = data$city

library("sp")

# Загрузка точек без указания проекции

pts <- SpatialPoints(sp)

pts

plot(pts)

# Загрузка точек с указанием проекции WGS84

pts <- SpatialPoints(sp, crs84)

pts

plot(pts)

# Преобразование проекции в WGS84 Russia

pts\_ru <- spTransform(pts, crsRU)

pts\_ru

plot(pts\_ru)

#### Загрузка данных из массива данных

sdf <- data

coordinates(sdf) <- ~lon + lat

proj4string(sdf) <- crs84

sdf\_ru <- spTransform(sdf, crsRU)

plot(sdf\_ru)

# Проверка отображения данных на карте России

library(sf)

# Загрузка файла ru формата ESRI

map <- st\_read("ru.shp")

# Установка проекции в WSG84 Russia

map <- st\_transform(map, crs=crsRU)

# Выбор только карты

map\_ru <- map[0]

# Отображение карты

plot(map\_ru)

# Отображение точек pts\_ru

points(pts\_ru, col='red')

# Отображение точек sdf\_ru

points(sdf\_ru, col='blue')

#################################

### Работа с временными данными

library("spacetime")

library("xts")

# Перечень годов для заданного набора данных

years <- 15:19

# Имя переменной для анализа

v <- 'IfoORT'

### Подготовка набора данных для выбранной переменной

# Список полей

fields <- paste(v,years,sep='\_')

fields

# Создание массива данных только из этих полей

df0 <- data[,fields]

head(df0)

# Преобразование в матрицу (без указания имён полей)

m <- as.matrix(df0, use.names=FALSE)

head(m)

# Транспонирование

m\_t <- t(m)

head(m\_t)

# Перевод из матрицы в вектор

values <- c(m\_t)

head(values)

# Создание из вектора набора данных

df <- data.frame(values=values)

names(df) <- c(v)

head(df)

# Ручное создание вектора временных меток

tm <-

c(

as.POSIXct("2015-01-01"),

as.POSIXct("2016-01-01"),

as.POSIXct("2017-01-01"),

as.POSIXct("2018-01-01"),

as.POSIXct("2019-01-01")

)

tm

# Автоматическое создание вектора временных меток

tm <- xts(years, as.POSIXct(paste('20', years, '-01-01', sep='')))

tm

# Создание STFDF (spatio-temporal data with full space-time grid)

stfdf <- STFDF(pts, time=tm, data=df)

stplot(stfdf)

# Режим вывода "xy" maps for time steps are plotted (по-умолчанию)

stplot(stfdf, mode='xy')

# Режим вывода "xt", a space-time plot is constructed

stplot(stfdf, mode='xt')

# Режим вывода по меткам (город)

stplot(stfdf, mode='xt', scaleX=0)

# Режим вывода по долготе

stplot(stfdf, mode='xt', scaleX=1)

# Режим вывода по широте

stplot(stfdf, mode='xt', scaleX=2)

# Режим вывода "ts", multiple-locations time series are plotted in a single plot, or in a separate panel for each attribute;

stplot(stfdf, mode='ts')

# Сгладим значения с помощью логарифмической функции

stfdf <- STFDF(pts, time=tm, data=log(df))

stplot(stfdf)

# Режим вывода "xt", a space-time plot is constructed

stplot(stfdf, mode='xt')

# Режим вывода по меткам

stplot(stfdf, mode='xt', scaleX=0)

# Режим вывода по долготе

stplot(stfdf, mode='xt', scaleX=1)

# Режим вывода по широте

stplot(stfdf, mode='xt', scaleX=2)

# Режим вывода "ts", multiple-locations time series are plotted in a single plot, or in a separate panel for each attribute;

stplot(stfdf, mode='ts')

# Режим вывода "tp" для каждой метки

stplot(stfdf, mode='tp')

# Работа с выборкой отдельных данных из STFDF

# Первый аргумент - пространственные метки, второй аргумент - временные точки

stplot(stfdf[, 0:3])

stfdf['Москва']

plot(stfdf['Москва'])

stfdf['Москва', 0:3]

plot(stfdf['Москва', 0:3])

stplot(stfdf[c('Москва', 'Санкт-Петербург', 'Новосибирск', 'Владивосток'), 0:3])