IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 7 Issue 6, June 2020
ISSN (Online) 2348 – 7968 | Impact Factor (2020) – 6.72
www.ijiset.com

Geostatistical Modeling Using Open Source R Software

IGN Mindra Jaya[1], Neneng Sunengsih[2]
[1, 2]Department Statistics, Padjadjaran University, Indonesia
*email: mindra@unpad.ac.id

Abstract

In recent years, the use of Geostatistical modeling has greatly increased. It provides a method for processing spatial and temporal data. In this study, the geostatistical modeling is used as a tool to modeling spatial continuity in order to describe climate phenomenon. R is open source software that is usually used for statistical computation. We provide a brief introduction to R and several geostatistical packages for continuous interpolation. There are two main groups of interpolation techniques, deterministic and geostatistical. The deterministic model for example inverse distance and geostatistical modeling such as kriging. We use data from the worlClim website and introduce ways to extract climate data from worldClim.

Keywords: geostatistics, R, Kriging, spatial prediction, spatial data analysis, woldClim.

1. Introduction

Geostatistical model is a statistical technique which provides method for processing spatial and temporal point data [1] [2]. Geostatistical analyst uses sample points taken at a different location in space and/or time to construct continuous maps. The sample points may describe some phenomena, such as climate variation including minimum, maximum and average temperatures, precipitation, solar duration, evaporation, and the other environmental phenomenon. Geostatistical Analyst provides high-resolution surface maps using the observed values from the measured locations to predict values for un-sample locations [3]. Geostatistical Analyst provides two groups of interpolation techniques: deterministic and stochastic [4]. The basic idea on Geostatistical modeling based on Tobler's First Law of Geography [5] Everything is related to everything else. But near things are more related than distant things. Deterministic techniques use mathematical functions for interpolation and stochastics rely on both statistical and

mathematical methods, which can be used to create surfaces and assess the uncertainty of the predictions.

R is an open source software for statistical computing and graphics. It can be run on a wide variety of UNIX platforms, Windows and MacOS. The use of R-software have been increased significantly because it provides many packages needed by researchers. There are several packages which are related to geostatistical analysis including `gstat` [6] and `georob` [7]. However, for some practitioners, working with R is not always easy. A lot of code that has to be recorded but not all practitioners have a good documentation. Here we provide some examples and code for geostatistical interpolation analysis especially for deterministic and geostatistic approaches. In order to provide a simple example with real data set, we consider to use climate data set from worldClim website and provides a way to extract the data using R-packages.

The rest of the paper consist introduction of R software in section 2, Geostatistical modeling in section 3 and section 4 provides discussion and conclusion.

2. R-software

There are many commercial software for implementing geostatistical analysis such as ArcGIS software and MAPINFO. R is an open source software and free for any user. R is multi-platform and easy to learn with a lot of packages that cover a lot of statistical problems [2]. The R-software can be downloaded at https://cran.r-project.org/ [8]. Below the basic syntax that need to be noted:

```
help.start()             #Load on line HTML help
help(function)           #Show on line help for "function"
?function                #Show on line help for"function"
help.search("keyword")   #Open RGui dialog for
```

```
                          packages/functions
                        #...or classes connected to "keyword"
q()                     #Quit R
library()               #List downloaded libraries
library(package)        #Load "package"
install.packages(package)#Download "package"
```

3. Geostatistical analysis

Geostatistical analysis provides two groups of interpolation techniques: deterministic and geostatistical.

3.1. Deterministic: Inverse Distance Weighted

Inverse distance weighting extends the basic idea of using similarity and attempts to quantify it. Instead only value at the nearest data location being used, values at other data locations are used but are weighted. The weight for each data location is assumed to be inversely related to the distance between that location and the location where an estimate is desired. The estimator is the form [9]:

$$\hat{z}_o = \frac{\sum_{j=1}^{m} k_j z_j}{\sum_{j=1}^{m} k_j} \quad (1)$$

where z_j are from the "m" closest positions and the weights k_j are chosen larger for data value near to where the estimate $\hat{z}_o$ is to be made. The denominator is a normalizing factor.

3.2. Geostatistical: Kriging

Kriging has been introduced by Matheron (1971) [10]. The kriging estimator is similar to the multiple regression estimator. The basic idea of kriging is combining variogram random function theory. The form of estimate $\hat{z}_o$ of the attribute at unmeasured location is presented below [9]:

$$\hat{z}_o = \sum_{j=1}^{m} \lambda_j z_j \tag{2}$$

The unbiased estimator of λ_j can be reached by considering constraint sum of λ_j to be 1:

$$\sum_{j=1}^{m} \lambda_j = 1 \tag{3}$$

There is one think left to determine for the prediction -the vector weights [1]:

$$\begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \\ \mu \end{bmatrix} = \begin{bmatrix} C_{11} & \dots & C_{1n} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ C_{n1} & \dots & C_{nn} & 1 \\ 1 & \dots & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} C_{10} \\ \vdots \\ C_{n0} \\ 1 \end{bmatrix} \tag{4}$$

Where μ is a Lagrange parameter and C_{ij} is a covariance between z_i and z_j. A relationship between a covariance and a variogram is following:

$$C_{ij} = Cov(z_i, z_j) = C(0) - \gamma(i - j) \tag{5}$$

where $C(0)$ is the *sill* of the variogram model.

4. Result and Discussion

The basic idea of this paper is providing the step by step procedures in analyzing spatial point data by means Geostatistical modeling. We use climate variables data from worldClim database. In this section we will introduced the simple way in which we can get climate data, by using WorldClim global climate data set. worldClim is a group of global gridded climate data, including temperature (mean, max, min), precipitation, solar radiation, wind speed and water vapour pressure. WorldClim contains monthly averages for the period between 1970 and 2018

with resolution in grid squares representing an area ~1 km2. The data are based on weather station data [11] [12].

Packages R are required:

- library(raster)
- library(sp)
- library(rgeos)
- library(maptools)
- library(splancs)
- library(ggplot2)
- library(gstat)
- library(sf)
- library(dplyr)
- library(tidyverse)

4.1. Downloading worldClim data

The climate data can be downloaded at website: https://www.worldclim.org/data/worldclim21.html. The data is available at the four spatial resolutions, between 30 seconds (~1 km^2) to 10 minutes (~340 km^2). Each download is a "zip" file containing 12 GeoTiff (.tif) files, one for each month of the year (January is 1; December is 12).

Table 1. The average climate data

variable	10 minutes	5 minutes	2.5 minutes	30 seconds
minimum temperature (°C)	tmin 10m	tmin 5m	tmin 2.5m	tmin 30s
maximum temperature (°C)	tmax 10m	tmax 5m	tmax 2.5m	tmax 30s
average temperature (°C)	tavg 10m	tavg 5m	tavg 2.5m	tavg 30s
precipitation (mm)	prec 10m	prec 5m	prec 2.5m	prec 30s
solar radiation (kJ m^{-2} day^{-1})	srad 10m	srad 5m	srad 2.5m	srad 30s
wind speed (m s^{-1})	wind 10m	wind 5m	wind 2.5m	wind 30s
water vapor pressure (kPa)	vapr 10m	vapr 5m	vapr 2.5m	vapr 30s

Source: https://www.worldclim.org

From this page you can download historical monthly weather data for 1960-2018. These data are downscaled from CRU-TS-4.03 by the Climatic Research Unit, University of East Anglia,

using WorldClim 2.1 for bias correction. The variables available are average minimum temperature (°C), average maximum temperature (°C) and total precipitation (mm). The spatial resolution is 2.5 minutes (~21 km2). Each download is a "zip" file containing 120 GeoTiff (.tif) files, for each month of the year (January is 1; December is 12), for a 10 year period.

Table 2. The climate data

years	minimum temperature	maximum temperature	precipitation
1960-1969	tmin_1960-1969	tmax_1960-1969	prec_1960-1969
1970-1979	tmin_1970-1979	tmax_1970-1979	prec_1970-1979
1980-1989	tmin_1980-1989	tmax_1980-1989	prec_1980-1989
1990-1999	tmin_1990-1999	tmax_1990-1999	prec_1990-1999
2000-2009	tmin_2000-2009	tmax_2000-2009	prec_2000-2009
2010-2018	tmin_2010-2018	tmax_2010-2018	prec_2010-2018

Source: https://www.worldclim.org

4.2. Extracting WorldClim data

To work with GeoTiff files in R, we'll once again use the "raster" package. If you have not already installed this package, go ahead and install.packages("raster") to install it. And remember to load the package after installation library(raster). For this guide, we are going to load all the data into R by creating separate raster objects: *RasterLayers*, for each month.

For example we extract the precipitation data

Let assume we already downloaded the data from https://www.worldclim.org/ and save at the Folder "F:/CLIMATE". Let we extracted the precipitation data for the coordinates below:

Table 3. Latitude and Longitude Example

ID	Latitude (x)	Longitude (y)
1	107.580	-6.911
2	107.661	-6.917
3	107.677	-6.920
4	107.602	-6.934
5	107.578	-6.944

The R-code

```
x<-c(107.580, 107.661, 107.677, 107.602, 107.578)
y<-c(-6.911, -6.917, -6.920, -6.934, -6.944)
site <- c(1:5)
samples <- data.frame(site, x, y, row.names="site")
prec<-raster("wc2.1_30s_prec_01.tif")
prec.data<- samples
prec.data$Jan <- extract(prec, samples)
```

The output

```
prec.data
    x       y    Jan
1 107.580 -6.911 246
2 107.661 -6.917 266
3 107.677 -6.920 268
4 107.602 -6.934 255
5 107.578 -6.944 255
```

To map the precipitation output, we need shape file map. We can get it by using raster package in R.

4.3. Creating shape file using R

Some researchers who do not expert in GIS software may have a problem to get the shapefile maps. Although some website provides this kind of maps for free. However, sometimes the maps that we need do not available. Using package `raster` we can get shapefile maps easily. Pakcage raster provides maps which including country (`level=0`), province (`level=1`), city (`level=2`), district (`level=3`), and village (`level=4`). For example we are going to crate shapefile maps for Bandung city, Indonesia including districts.

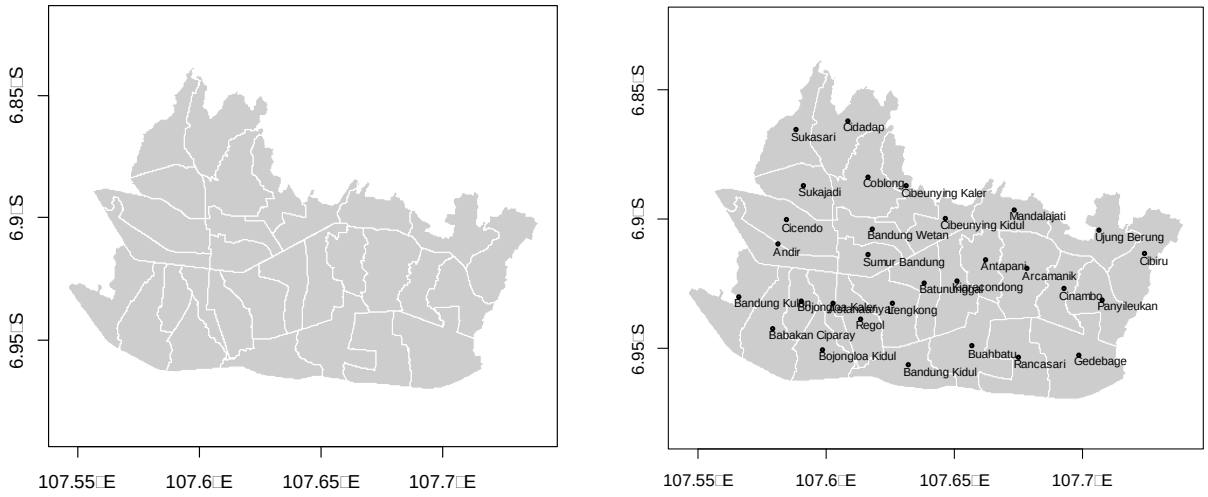
```
Indonesia<-getData('GADM', country='IDN', level=3)   #Get the Province


Shapefile for Indonesia
Bandung<- Indonesia[Indonesia $NAME_2 == "Kota Bandung",] #Get the Bandung
city Shapefile with districts
```

`getData` is a function to call GADM (Global Administration Boundaries) or woldclim data.

`country` is a function to define the country. For Indonesia the code is IDN. To visualize the map we can use function plot.

```
plot(Bandung, axes=TRUE,col="gray80",border="white")# (a) map without label
plot(Bandung, axes=TRUE,col="gray80",border="white")# (b) map with label
DistrictsName<-Bandung$NAME_3            #Get the names of the districts
Coordinates<-coordinates(Bandung)        #Find the centroid of each district
text(Coordinates-0.001, label=DistrictsName, cex=0.7) #Add label on the map
points(Coordinates+0.001,lwd=2,cex=0.5)   #Add points on the map
```

We can simply add the label name of the districts in Bandung city using code below:



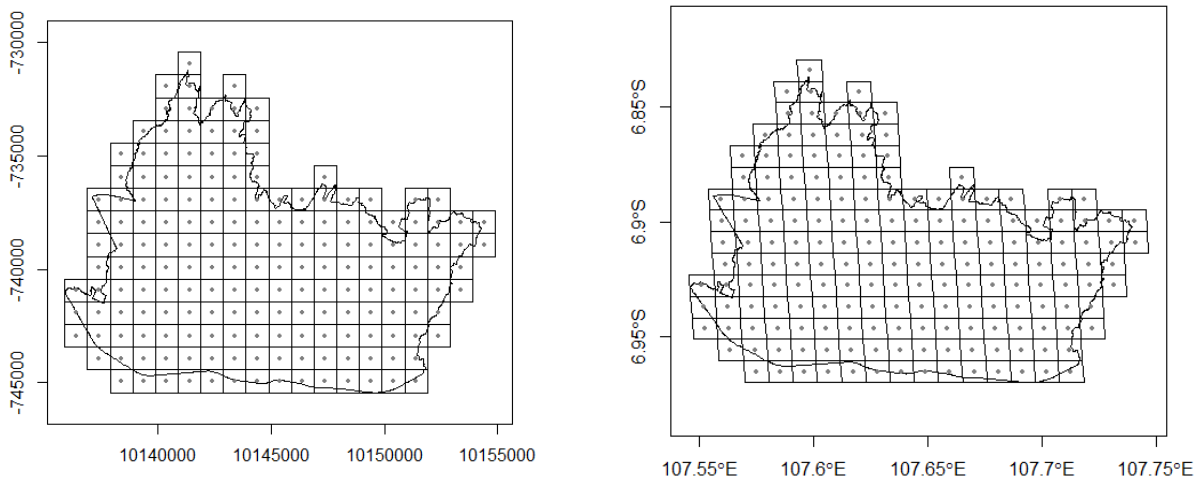
(a) Map without label (b) Map with label

Figure 1. Bandung city Map

Sometime we need to map the climate variables in appropriate fine-scale for example in grid-cell with resolution ~1 km^2. We can use following code to produce grid-cell in resolution ~1 km^2

```
Indonesia<-getData('GADM',  country='IDN',  level=2)    ##Get  the  Province
Shapefile for Indo
Bandung<-Indonesia[Indonesia$NAME_2 == "Kota Bandung",]

#Transformaxg vector data to UTM

UTM <- CRS("+proj=robin +datum=WGS84") #Transform to meters
BandungUTM <- spTransform(Bandung, UTM)
BandungUTMSf<-st_as_sf(BandungUTM)
```

```
GridCell<- 1000  # size of squares, in units of the CRS (i.e. meters for
BandungUTM 1000 M)

GridPolygonUTM <- st_make_grid(BandungUTMSf, square = T, cellsize =
c(GridCell, GridCell)) %>% # the grid, covering bounding box
               st_sf() # not really required, but makes the grid nicer to
work with later

###Find GridPoint
GridPolygon<-as(GridPolygonUTM, 'Spatial')
CoordUTM<-coordinates(GridPolygon)
Coord<-        coordinates(spTransform(GridPolygon,        CRS("+proj=longlat
+datum=WGS84")))

###Plot
plot(GridPolygonUTM, col = 'white', axes=T)
plot(st_geometry(BandungUTMSf), add = T)
points(CoordUTM,pch=19,col="gray50",cex=0.5)

####
GridPolygon1<-spTransform(GridPolygon, CRS("+proj=longlat +datum=WGS84"))
plot(GridPolygon1, col = 'white', axes=T)
plot(Bandung, add=T)
points(Coord,pch=19,col="gray50",cex=0.5)
```



(a) UTM Coordinates (unit: m) (b) Degrees ^{0}E

Figure 2. Grid cell map for resolution (~ 1 km^2)

Next, we present the code to perform precipitation data from January-December on Bandung city map.

```
n<-nrow(CoordBandung)
x<-CoordBandung[,1]
```

```
y<-CoordBandung[,2]
site<-c(1:n)
samples <- data.frame(site, x, y, row.names="site")

#extract precipitation data for Bandung city
setwd("F:/PAPER 3/wc2.1_30s_prec")
prec1<-raster("wc2.1_30s_prec_01.tif")
prec2<-raster("wc2.1_30s_prec_02.tif")
prec3<-raster("wc2.1_30s_prec_03.tif")
prec4<-raster("wc2.1_30s_prec_04.tif")
prec5<-raster("wc2.1_30s_prec_05.tif")
prec6<-raster("wc2.1_30s_prec_06.tif")
prec7<-raster("wc2.1_30s_prec_07.tif")
prec8<-raster("wc2.1_30s_prec_08.tif")
prec9<-raster("wc2.1_30s_prec_09.tif")
prec10<-raster("wc2.1_30s_prec_10.tif")
prec11<-raster("wc2.1_30s_prec_11.tif")
prec12<-raster("wc2.1_30s_prec_12.tif")

prec.data <- samples
prec.data$A <- extract(prec1, samples)
prec.data$B <- extract(prec2, samples)
prec.data$C <- extract(prec3, samples)
prec.data$D <- extract(prec4, samples)
prec.data$E <- extract(prec5, samples)
prec.data$F <- extract(prec6, samples)
prec.data$G <- extract(prec7, samples)
prec.data$H <- extract(prec8, samples)
prec.data$I <- extract(prec9, samples)
prec.data$J <- extract(prec10, samples)
prec.data$K <- extract(prec11, samples)
prec.data$L <- extract(prec12, samples)
```

With the output are presented below.

```
> head(prec.data)
         x        y   A   B   C   D   E  F  G  H   I   J   K   L
1 107.5804 -6.910823 246 209 269 268 182 98 68 62 101 182 292 277
2 107.6612 -6.916857 266 223 285 272 178 88 64 58  87 170 281 290
3 107.6771 -6.920299 268 224 288 272 175 86 62 56  84 167 277 289
4 107.6017 -6.933670 255 214 277 271 180 94 67 62  97 182 290 284
5 107.5784 -6.943537 255 212 276 272 181 95 68 63 100 187 294 285
6 107.6312 -6.957653 260 217 283 273 177 88 63 59  92 179 288 289
```

Note x: latitude; y: longitude A-L: January-December

```
prec.data1<-prec.data
coordinates(prec.data1)<-~x+y
```

```
Month_names <- c(
                    `A` = "January",
                    `B` = "February",
                    `C` = "March",
                    `D` = "April",
                    `E` = "May",
                    `F` = "June",
                    `G` = "July",
                    `H` = "August",
                    `I` = "September",
                    `J` = "October",
                    `K` = "November",
                    `L` = "December"
                    )
prec.data1 %>% as.data.frame %>%
gather(Month, Precipitation, A:L) %>%
ggplot(aes(x, y)) +
  coord_fixed(ratio = 1)+
  facet_wrap(.~Month,labeller = as_labeller(Month_names), ncol=3)+
  geom_point(aes(size=Precipitation), color="blue", alpha=3/4) +
  coord_equal() + theme_bw()+
  geom_map(data = Bandung1,map =Bandung1,
aes(x=long,y=lat,map_id=id),fill="white",alpha=0.0,color="gray50",size=0.1) +
   theme(axis.text.x = element_text(angle = 90))+
  labs(fill = "Precipitation")+ theme(legend.position="bottom")+ theme(text =
element_text(size=10))  +
  guides(fill = guide_colorbar(title.position = "left", title.vjust = 1,
                               frame.colour = "black",
                               barwidth = 15,
                               barheight = 1.7))
```

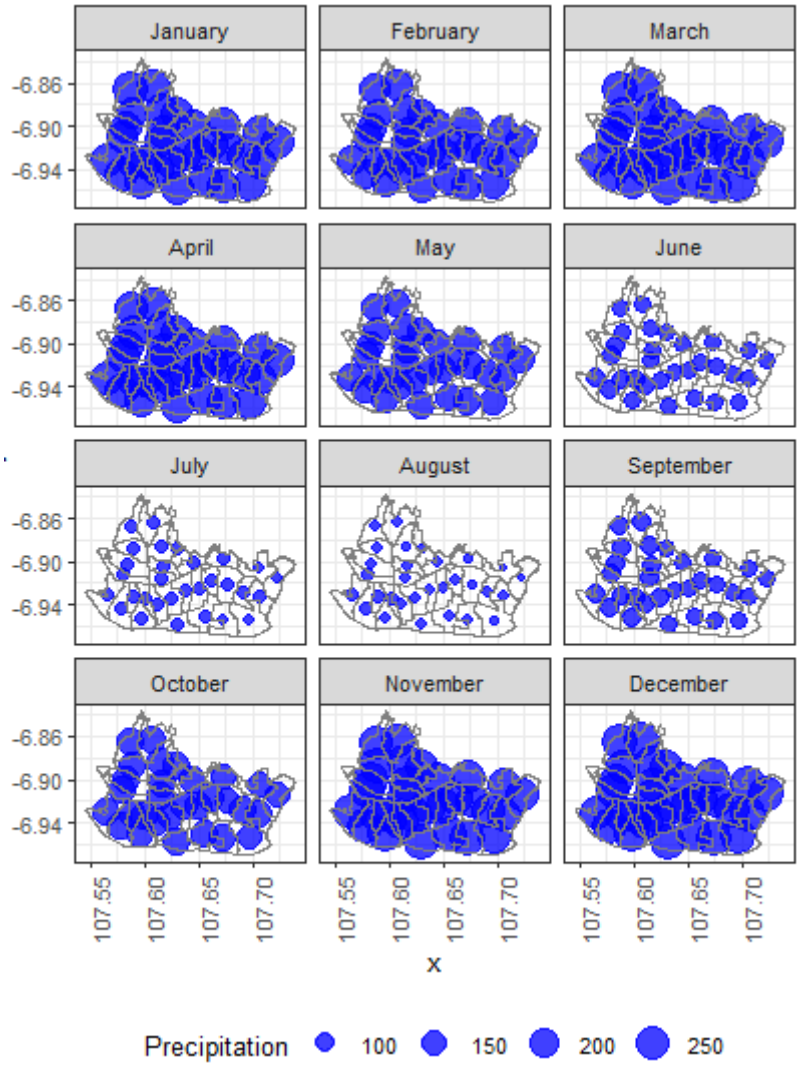


Figure 3. Point Unit Observations

4.4 Inverse Distance (IDW)

In order to obtain the continuous interpolation map, we can use IDW procedure with the detail code are presented below.

```
Indo<-getData('GADM', country='IDN', level=2)  ##Get the
Province Shapefile for Indo
Bdg<-Indo[Indo$NAME_2 == "Kota Bandung",]

#1. Make a rectangular grid over your SpatialPolygonsDataFrame
grd <- makegrid(Bdg, n = 10000)
colnames(grd) <- c('x','y')
```

```
#2. Get the outline of your iPRegularly shaped polygon
outline <- Bdg@polygons[[1]]@Polygons[[2]]@coords

#3. Use inout() for clipping the rectangular grid by outline
new_grd <- grd[inout(grd,outline), ]

#4. Visualize your clipped grid, which can be used for kriging!
ggplot(new_grd) +geom_point(aes(x=x,y=y))

new_grd <- grd[inout(grd,outline), ]
coordinates(new_grd)<-~x+y
gridded(new_grd) = TRUE

coordinates(prec.data)<-~x+y

prec.data.Jan = krige(prec.data$A ~ 1, prec.data, new_grd)
prec.data.Feb = krige(prec.data$B ~ 1, prec.data, new_grd)
prec.data.Mar = krige(prec.data$C ~ 1, prec.data, new_grd)
prec.data.Apr = krige(prec.data$D ~ 1, prec.data, new_grd)
prec.data.May = krige(prec.data$E ~ 1, prec.data, new_grd)
prec.data.Jun = krige(prec.data$F ~ 1, prec.data, new_grd)
prec.data.Jul = krige(prec.data$G ~ 1, prec.data, new_grd)
prec.data.Aug = krige(prec.data$H ~ 1, prec.data, new_grd)
prec.data.Sep = krige(prec.data$I ~ 1, prec.data, new_grd)
prec.data.Oct = krige(prec.data$J ~ 1, prec.data, new_grd)
prec.data.Nov = krige(prec.data$K ~ 1, prec.data, new_grd)
prec.data.Dec = krige(prec.data$L ~ 1, prec.data, new_grd)

prec.dataS<-prec.data.Jan[,-c(1,2)]
prec.dataS$A<-prec.data.Jan$var1.pred
prec.dataS$B<-prec.data.Feb$var1.pred
prec.dataS$C<-prec.data.Mar$var1.pred
prec.dataS$D<-prec.data.Apr$var1.pred
prec.dataS$E<-prec.data.May$var1.pred
prec.dataS$F<-prec.data.Jun$var1.pred
prec.dataS$G<-prec.data.Jul$var1.pred
prec.dataS$H<-prec.data.Aug$var1.pred
prec.dataS$I<-prec.data.Sep$var1.pred
prec.dataS$J<-prec.data.Oct$var1.pred
prec.dataS$K<-prec.data.Nov$var1.pred
prec.dataS$L<-prec.data.Dec$var1.pred

Month_names <- c(
                    `A` = "January",
                    `B` = "February",
                    `C` = "March",
                    `D` = "April",
```

```
                        `E` = "May",
                        `F` = "June",
                        `G` = "July",
                        `H` = "August",
                        `I` = "September",
                        `J` = "October",
                        `K` = "November",
                        `L` = "December"
                        )

#Online version
P<-  prec.dataS %>% as.data.frame %>%
  gather(Month, Precipitation, A:L) %>%
  ggplot(aes(x=x, y=y)) + geom_tile(aes(fill=Precipitation)) +
  facet_wrap(.~Month,labeller = as_labeller(Month_names),
ncol=4)+ coord_fixed(ratio = 1) +
  scale_fill_gradientn(colours = rev(rainbow(7))) +
theme_bw()+ylab("Latitude")+xlab("Longitude")+
  labs(fill = "Prepicipitation")+
theme(legend.position="bottom")+ theme(text =
element_text(size=12))  +
  guides(fill = guide_colorbar(title.position = "left",
title.vjust = 1,
                               frame.colour = "black",
                               barwidth = 15,
                               barheight = 1.7))


P+geom_map(data = BdgData,map =BdgData,
aes(x=long,y=lat,map_id=id),fill="white",alpha=0.0,color="white"
,size=0.1)
```

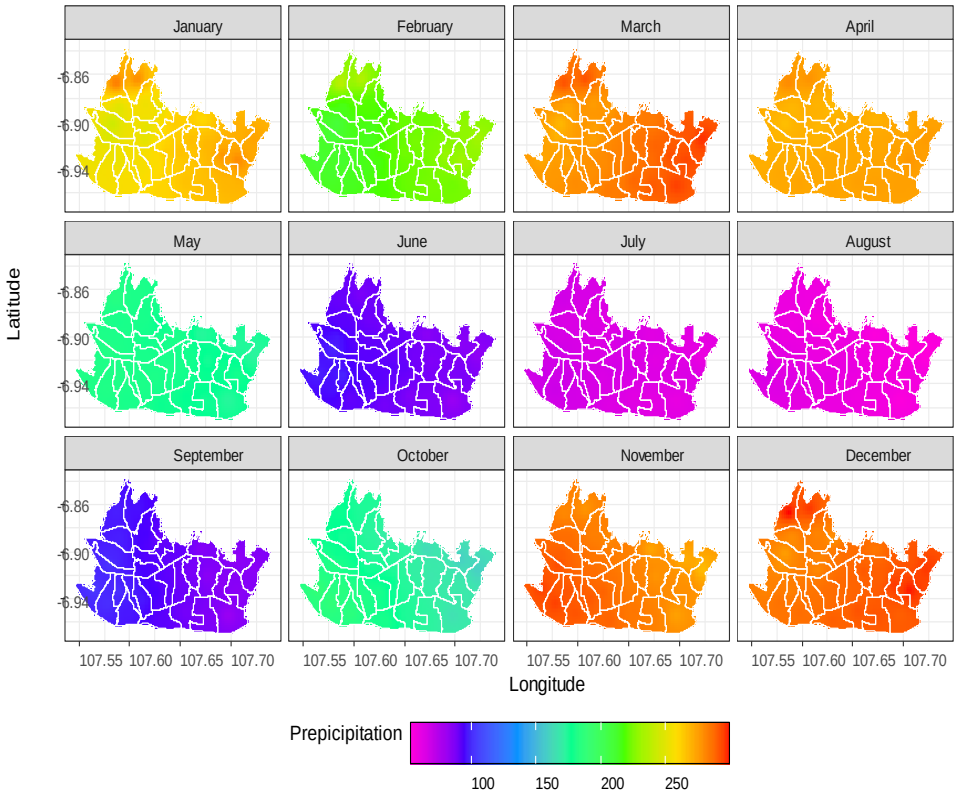


Figure 4. IDW Maps of Precipitation on Bandung City

4.5 Kriging

Below we present the code for Kriging.

```
####Krging

coordinates(prec.data1) %>% glimpse
proj4string(prec.data1)
identical( bbox(prec.data1), prec.data1@bbox )
identical( coordinates(prec.data1), prec.data1@coords )
prec.data1@data %>% glimpse
prec.data1 %>% as.data.frame %>% glimpse


A <- variogram(A~1, prec.data1) # calculates sample variogram values
A.fit <- fit.variogram(A, model=vgm("Sph")) # fit model

B <- variogram(B~1, prec.data1) # calculates sample variogram values
B.fit <- fit.variogram(B, model=vgm("Sph")) # fit model

C <- variogram(C~1, prec.data1) # calculates sample variogram values
C.fit <- fit.variogram(C, model=vgm("Sph")) # fit model

D <- variogram(D~1, prec.data1) # calculates sample variogram values
```

```
D.fit <- fit.variogram(D, model=vgm("Sph")) # fit model

E <- variogram(E~1, prec.data1) # calculates sample variogram values
E.fit <- fit.variogram(E, model=vgm("Sph")) # fit model

F <- variogram(F~1, prec.data1) # calculates sample variogram values
F.fit <- fit.variogram(F, model=vgm("Sph")) # fit model

G <- variogram(G~1, prec.data1) # calculates sample variogram values
G.fit <- fit.variogram(G, model=vgm("Sph")) # fit model

H <- variogram(H~1, prec.data1) # calculates sample variogram values
H.fit <- fit.variogram(H, model=vgm("Sph")) # fit model

I <- variogram(I~1, prec.data1) # calculates sample variogram values
I.fit <- fit.variogram(I, model=vgm("Sph")) # fit model

J <- variogram(J~1, prec.data1) # calculates sample variogram values
J.fit <- fit.variogram(J, model=vgm("Sph")) # fit model

K <- variogram(K~1, prec.data1) # calculates sample variogram values
K.fit <- fit.variogram(K, model=vgm("Sph")) # fit model

L <- variogram(L~1, prec.data1) # calculates sample variogram values
L.fit <- fit.variogram(L, model=vgm("Sph")) # fit model
```

To evaluate the goodness of fit between empirical and theoretical variogram, we can provides plot below.

```
#present the sample and fit variogram
plot(A, A.fit,main="January")
plot(B, B.fit,main="February")
plot(C, C.fit,main="March")
plot(D, D.fit,main="April")
plot(E, E.fit,main="May")
plot(F, F.fit,main="Jun")
plot(G, G.fit,main="July")
plot(H, H.fit,main="August")
plot(I, I.fit,main="September")
plot(J, J.fit,main="October")
plot(K, K.fit,main="November")
plot(L, L.fit,main="December")
```

The plots in Figure 5 show there are some months which have good fit and the other need to be improved. Here we use spherical model "`sph`" for all months. To improve the model, we can try to use the others model such as exponential ("`Exp`"), Gaussian ("`Gau`"), and Mattern ("`Mat`").

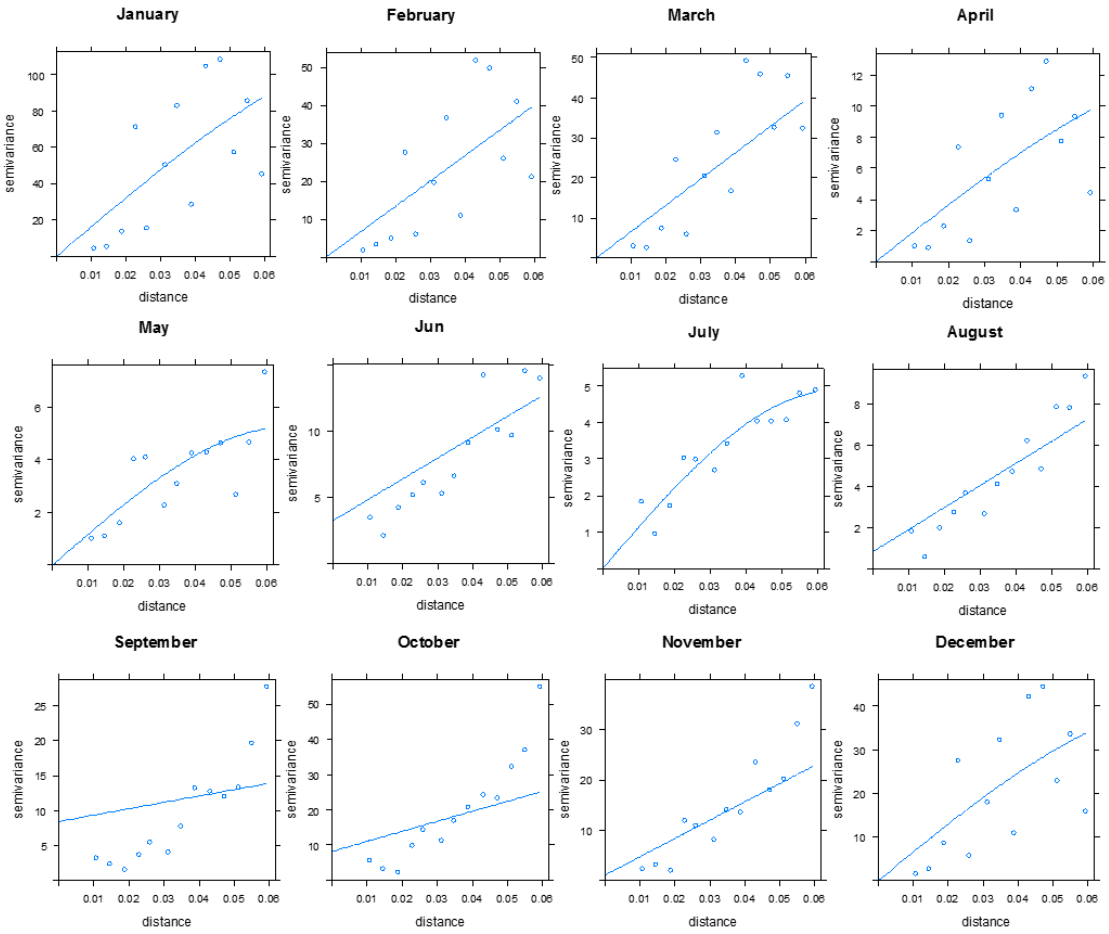
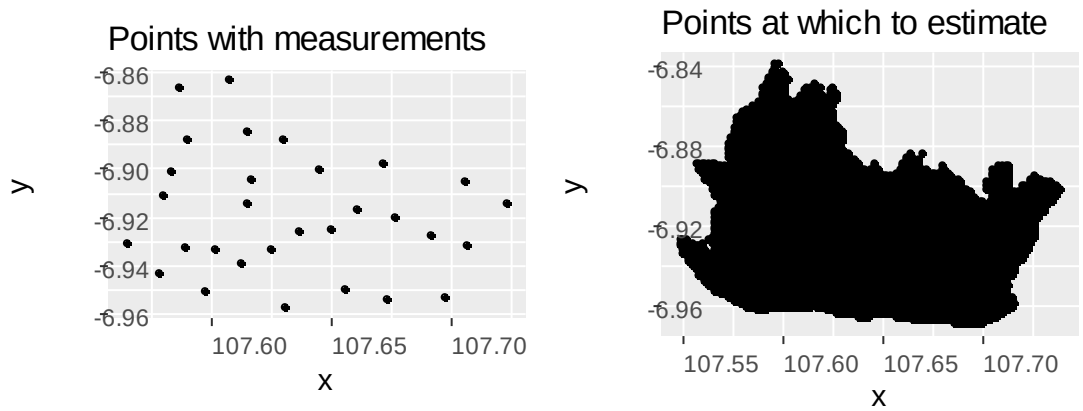


Figure 5. Empirical versus Theoretical Variogram

```
plot1 <- prec.data1 %>% as.data.frame %>%
  ggplot(aes(x, y)) + geom_point(size=1) + coord_equal() +
  ggtitle("Points with measurements")

# this is clearly gridded over the region of interest
plot2 <- new_grd %>% as.data.frame %>%
  ggplot(aes(x, y)) + geom_point(size=1) + coord_equal() +
  ggtitle("Points at which to estimate")

library(gridExtra)
grid.arrange(plot1, plot2, ncol = 2)

A.kriged <- krige(A ~ 1, prec.data1, new_grd, model=A.fit)
B.kriged <- krige(B ~ 1, prec.data1, new_grd, model=B.fit)
C.kriged <- krige(C ~ 1, prec.data1, new_grd, model=C.fit)
D.kriged <- krige(D ~ 1, prec.data1, new_grd, model=D.fit)
E.kriged <- krige(E ~ 1, prec.data1, new_grd, model=E.fit)
F.kriged <- krige(F ~ 1, prec.data1, new_grd, model=F.fit)
G.kriged <- krige(G ~ 1, prec.data1, new_grd, model=G.fit)
H.kriged <- krige(H ~ 1, prec.data1, new_grd, model=H.fit)
I.kriged <- krige(I ~ 1, prec.data1, new_grd, model=I.fit)
```

```
J.kriged <- krige(J ~ 1, prec.data1, new_grd, model=J.fit)
K.kriged <- krige(K ~ 1, prec.data1, new_grd, model=K.fit)
L.kriged <- krige(L ~ 1, prec.data1, new_grd, model=L.fit)

prec.kriged<-A.kriged[,-c(1,2)]
prec.kriged$A<-A.kriged$var1.pred
prec.kriged$B<-B.kriged$var1.pred
prec.kriged$C<-C.kriged$var1.pred
prec.kriged$D<-D.kriged$var1.pred
prec.kriged$E<-E.kriged$var1.pred
prec.kriged$F<-F.kriged$var1.pred
prec.kriged$G<-G.kriged$var1.pred
prec.kriged$H<-H.kriged$var1.pred
prec.kriged$I<-I.kriged$var1.pred
prec.kriged$J<-J.kriged$var1.pred
prec.kriged$K<-K.kriged$var1.pred
prec.kriged$L<-L.kriged$var1.pred

P2<-  prec.kriged %>% as.data.frame %>%
  gather(Month, Precipitation, A:L) %>%
  ggplot(aes(x=x, y=y)) + geom_tile(aes(fill=Precipitation)) +
  facet_wrap(.~Month,labeller = as_labeller(Month_names), ncol=4)+
coord_fixed(ratio = 1) +
  scale_fill_gradientn(colours = rev(rainbow(7))) +
theme_bw()+ylab("Latitude")+xlab("Longitude")+
  labs(fill = "Prepicipitation")+ theme(legend.position="bottom")+ theme(text
= element_text(size=12)) +
  theme(axis.text.x = element_text(angle = 90)) +
  guides(fill = guide_colorbar(title.position = "left", title.vjust = 1,
                               frame.colour = "black",
                               barwidth = 15,
                               barheight = 1.7))


P2+geom_map(data = Bandung1,map =Bandung1,
aes(x=long,y=lat,map_id=id),fill="white",alpha=0.0,color="white",size=0.1)
```



(a) Point units (b) Grid Points

Figure 6. Point and Grid

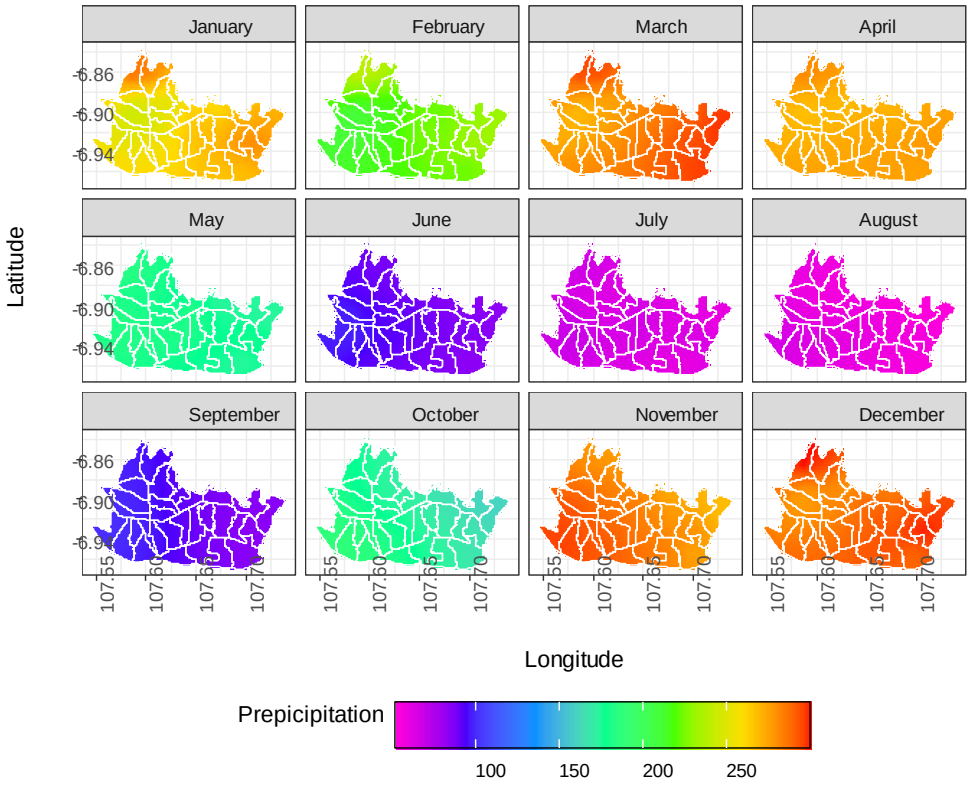


Figure 7. Kriging Prediction

The results of IDW and Co-Kriging look almost similar. The high precipitation can be found between November to May every year.

5. Conclusion

Geostatistical analysis has been used widely in environmental, ecology, epidemiology, climatology, and other disciplines. wordClim database provides very nice climates and bioclimatic variables at the four spatial resolutions, between 30 seconds (~1 km2) to 10 minutes (~340 km2) that can be used for any kind of research purposes and it is free for downloading. R-software facilitates geostatistical computation and free. This research provides procedure and R-code that can be used for practitioners who do know well the R-code for geostatistical modeling.

This will greatly help researchers in particular shorten the time of data analysis because each code needed in the analysis of the parameter estimation and mapping is presented in detail here and some code has never been published which is the creativity of the researcher. Finally, we believe that this will provide benefits in geostatistical modeling for various applications.

References

[1] A. Volfová and M. Šmejkal, "Geostatistical Methods in R," Geoinformatics FCE CTU, vol. 8, pp. 29-54, 2012.

[2] R. G. Maliva, "Geostatistical Methods and Applications," in Aquifer Characterization Techniques Schlumberger Methods in Water Resources Evaluation Series No. 4, Switzerland, Springer, 2016, pp. 595-617.

[3] M. Blangiardo and M. Cameletti, Spatial and Spatio-temporal Bayesian Models with R - INLA, Chichester: John Wiley & Sons, 2015.

[4] C. B. Vîlceanu, A. C. Badea and S. Herban, "Geostatistical Analysis for Landslide Prediction in Transilvania, Romania," The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vols. XLII-3, no. W4, pp. 543-547, 2018.

[5] W. Tobler, "A Computer Movie Simulating Urban Growth in the Detroit Region," Economic Geography, vol. 46, pp. 234-240, 1970.

[6] E. Pebesma, "Multivariable geostatistics in S: the gstat package," Computers & Geosciences, vol. 30, no. 7, pp. 683-691 , 2004.

[7] A. Papritz, "Tutorial and Manual for Geostatistical Analyses with the R package georob," 28 November 2016. [Online]. Available: https://mran.microsoft.com/snapshot/2016-12-03/web/packages/georob/vignettes/georob_vignette.pdf. [Accessed 10 May 2020].

[8] R-Team, "R-Project," R Foundation, 24 April 2020. [Online]. Available: https://www.r-project.org/. [Accessed 10 May 2020].

[9] J. Artiola, I. L. Pepper and M. L. Brusseau, Environmental Monitoring and Characterization, California: Elsevier Science & Technology Books, 2004.

[10] G. Matheron, The theory of regionalized variables and its applications. English translations of Les Cashiers du Centre de Morphologie Mathematique, Paris: ENSMP, 1971.

[11] Ben, "Extracting data and making climate maps using WorldClim datasets," benjaminbell, 26 January 2018. [Online]. Available: https://www.benjaminbell.co.uk/2018/01/extracting-data-and-making-climate-maps.html. [Accessed 8 May 2020].

[12] "worldclim," worldclim, 2020. [Online]. Available: https://www.worldclim.org/data/index.html. [Accessed 8 May 2020].