Kriging with external drift (KED)

Laura Delgado Bejarano
Agricultural Engineer, Universidad Nacional de Colombia
Master's Candidate, University of Campinas (UNICAMP)

2024-10-23

```r
knitr::opts_chunk$set(echo = TRUE)
# --- Directory settings (edit these paths as needed) ---
WORK_DIR <- "G:/Mi unidad/02_Maestria/01_Projeito/" # <- edit if needed
WORK_DIR <- normalizePath(WORK_DIR, winslash = "/", mustWork = TRUE)  # standardize slashes on Windows
knitr::opts_knit$set(root.dir = WORK_DIR)  # make this the working dir for all chunks
```

Libraries

Installing and loading the libraries that are going to be used

```r
# Package manager
if(!require(pacman)) install.packages("pacman")
```

```
## Loading required package: pacman
```

```r
# Load or install required packages
pacman::p_load(sp,gstat,ggspatial,rstudioapi,raster,
  ggplot2,gridExtra,ggstar,mapview,terra,sf,
  akima,automap,paletteer
)
```

Cleaning the space

```r
rm(list = ls())    # Clear all objects
graphics.off()     # Close graphics devices
cat("\014")        # Clear console
```

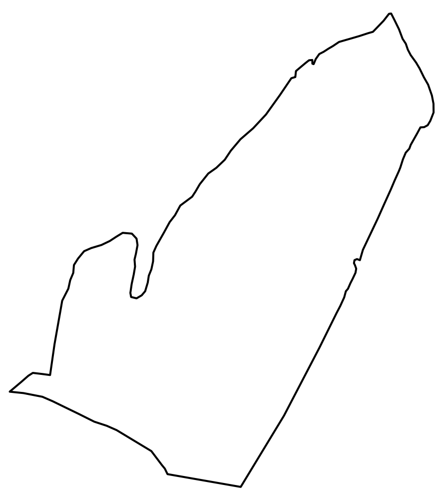
Loading the data

Load polygon of the area of study

```
poly <- sf::st_read("00_Cartobase/01_Contornos/01_Paulinia/Contorno_Paulinia.shp")
```

```
## Reading layer `Contorno_Paulinia' from data source
##   `G:\Mi unidad\02_Maestria\01_Projeito\00_Cartobase\01_Contornos\01_Paulinia\Contorno_Paulinia.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 1 feature and 0 fields
## Geometry type: POLYGON
## Dimension:     XYZ
## Bounding box:  xmin: 275598.9 ymin: 7487346 xmax: 277082.2 ymax: 7488999
## z_range:       zmin: 0 zmax: 0
## Projected CRS: WGS 84 / UTM zone 23S
```

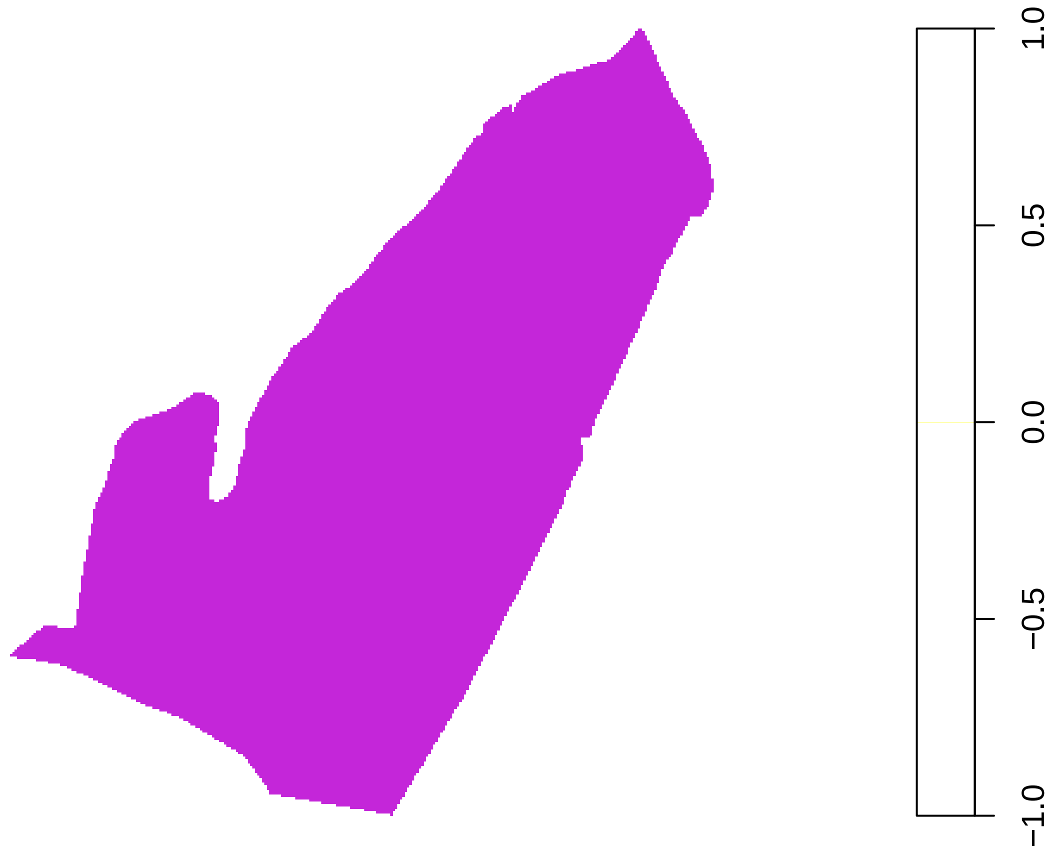
```
plot(poly)
```



```
poly_sf <- st_zm(poly)
```

Create a clean and empty grid to interpolate

```
r  <- raster(poly_sf, res = 5)          # base grid resolution (meters)
rp <- rasterize(poly_sf, r, 0)          # empty raster within polygon
grid <- as(rp, "SpatialPixelsDataFrame")
plot(grid)
```




```
proj4string(grid) <-CRS("+init=epsg:32723") #CRS area
```

```
## Warning in CPL_crs_from_input(x): GDAL Message 1: +init=epsg:XXXX syntax is
## deprecated. It might return a CRS with a non-EPSG compliant axis order.
```

```
proj4string(grid)
```

```
## [1] "+proj=utm +zone=23 +south +datum=WGS84 +units=m +no_defs"
```

Load the csv with the data

Here is important to know if the csv are separed by , or ; and change it if necessary

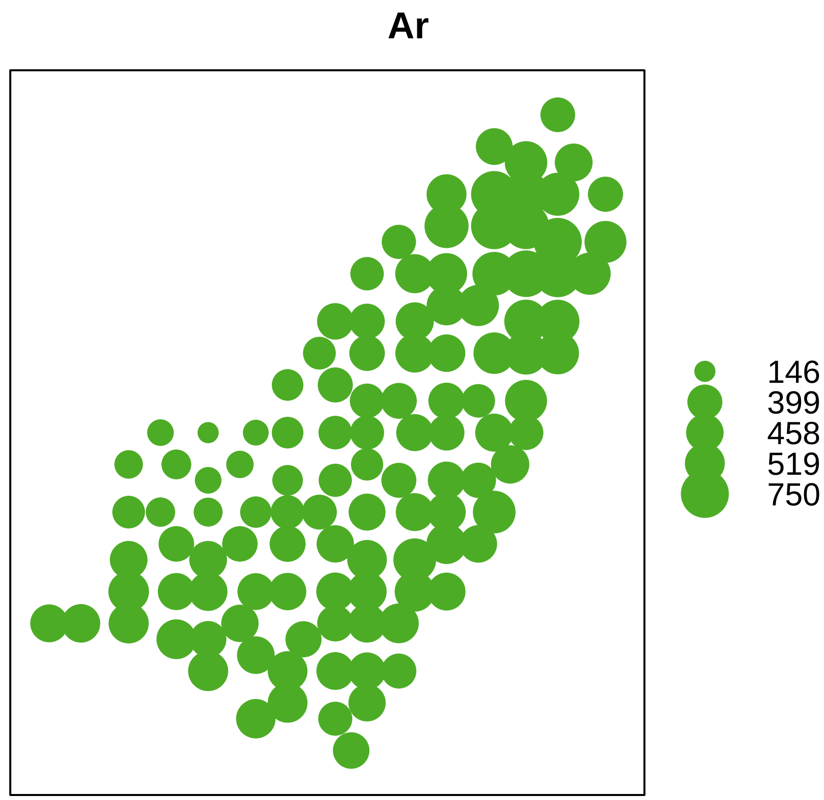
```r
original = data.frame(read.csv(file = "02_Cenarios_amostrais/04_CSVs/Paulinia_1am_cada1ha.csv",
 header = TRUE, sep = ';'))
 head(original)
```

```
##    pH  P   K CTC  V Argila        x       y
## 1 5.4 50 4.4  95 72    468 275619.8 7487681
## 2 5.8 75 4.6 102 78    484 275699.8 7487681
## 3 5.0 22 3.4  88 60    265 275819.8 7488081
## 4 4.6 18 2.8  70 53    348 275819.8 7487961
## 5 5.9 66 6.1 107 86    462 275819.8 7487841
## 6 5.5 51 5.0  94 78    537 275819.8 7487761
```

```r
 dados = original[,c(7,8,6)] # select columns of interest (x,y,z)
 head(dados)
```

```
##          x       y Argila
## 1 275619.8 7487681    468
## 2 275699.8 7487681    484
## 3 275819.8 7488081    265
## 4 275819.8 7487961    348
## 5 275819.8 7487841    462
## 6 275819.8 7487761    537
```

```r
 dados = na.omit(dados)
 names(dados) = c( "x", "y", "Ar") # ensure names x, y, variable
 sp::coordinates(dados) = ~x+y
 sp::bubble(dados, "Ar")
```

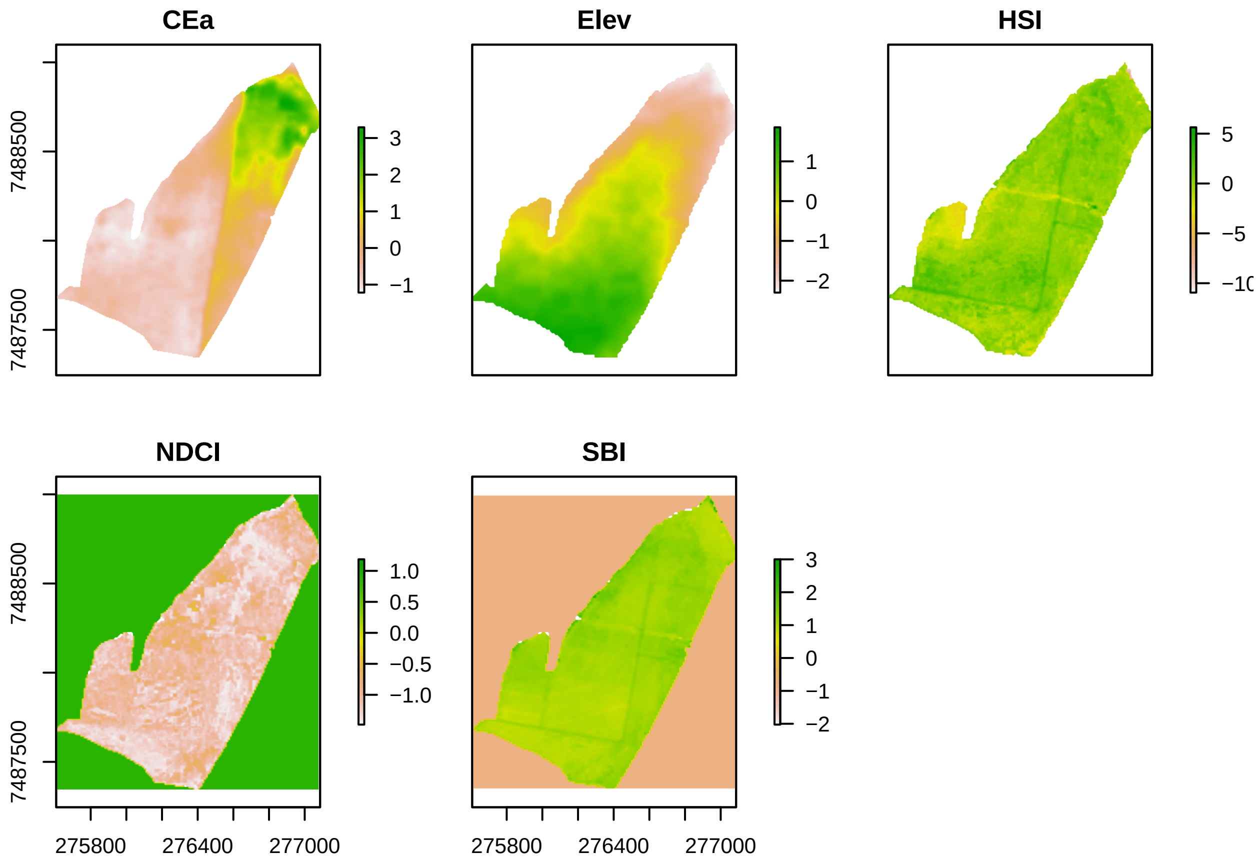


#Covariates

Load covariates and resample with bilinear method in the spatial resolution wanted

```
# List auxiliary rasters (.tif/.tiff)
covariates <- list.files(
  path = "01_Covariaveis/01_Paulinia/04_Stack",
  pattern = "\\.tif(f)?$",
  full.names = TRUE)

# Use elevation raster as template if available, otherwise the first raster
elev_idx <- grep("Elevacao_", basename(covariates), ignore.case = TRUE)
template_path <- if (length(elev_idx) > 0) covariates[elev_idx[1]] else covariates[1]
reference_raster <- raster::raster(template_path)
raster::res(reference_raster) <- 5

# Read, resample and standardize rasters
all_rasters <- lapply(covariates, raster::raster)
resampled   <- raster::stack(lapply(all_rasters, function(r) raster::resample(r, reference_raster, "bili
standardized <- raster::stack(lapply(1:raster::nlayers(resampled), function(i) {
  r <- resampled[[i]]
  v <- raster::getValues(r); ok <- is.finite(v)
  v[ok] <- as.vector(scale(v[ok], center = TRUE, scale = TRUE))
  r[] <- v; r
}))

names(standardized) <- c("CEa","Elev","HSI","NDCI","SBI")
```

```
plot(standardized)
```



##Extraction

Extraction of auxiliary information co-located with soil sampling points for data

```
Values<-raster::extract(standardized,dados)
head(Values)
```

```
##             CEa        Elev        HSI        NDCI         SBI
## [1,] -0.7914084  1.23828488  1.3505255 -1.09313653  0.09924985
## [2,] -0.6238500  1.29626377 -0.4498436 -1.29524827  0.96124230
## [3,] -0.9824437 -0.21308213 -1.7752143 -0.03109826 -0.10791708
## [4,] -0.7664722  0.01511799 -1.5038455 -1.23947384  1.19356536
## [5,] -0.6585380  0.71759496  0.9954052 -1.13764751  0.95769023
## [6,] -0.6003898  0.93851772  1.2164487 -1.04785119  0.82097464
```

```
dados@data<-cbind(dados@data,Values)
head(dados)
```

```
##    Ar        CEa        Elev        HSI        NDCI         SBI
## 1 468 -0.7914084  1.23828488  1.3505255 -1.09313653  0.09924985
## 2 484 -0.6238500  1.29626377 -0.4498436 -1.29524827  0.96124230
## 3 265 -0.9824437 -0.21308213 -1.7752143 -0.03109826 -0.10791708
## 4 348 -0.7664722  0.01511799 -1.5038455 -1.23947384  1.19356536
```

```
## 5 462 -0.6585380  0.71759496  0.9954052 -1.13764751  0.95769023
## 6 537 -0.6003898  0.93851772  1.2164487 -1.04785119  0.82097464
```

```r
#Combine with interpolation grid
grid_cov <- raster::extract(standardized, grid)
grid@data  <- cbind(grid@data, grid_cov) #the grid is going to have the value of the covariates to inte
```

Geostatistics

Adjust the experimental semivariogram, here is necessary to define the initial values based on the experimental semivariogram: vgm(psill (c), model, range (a), nugget (co))

The formula is going to have Z~Cov1+Cov2…+Covn

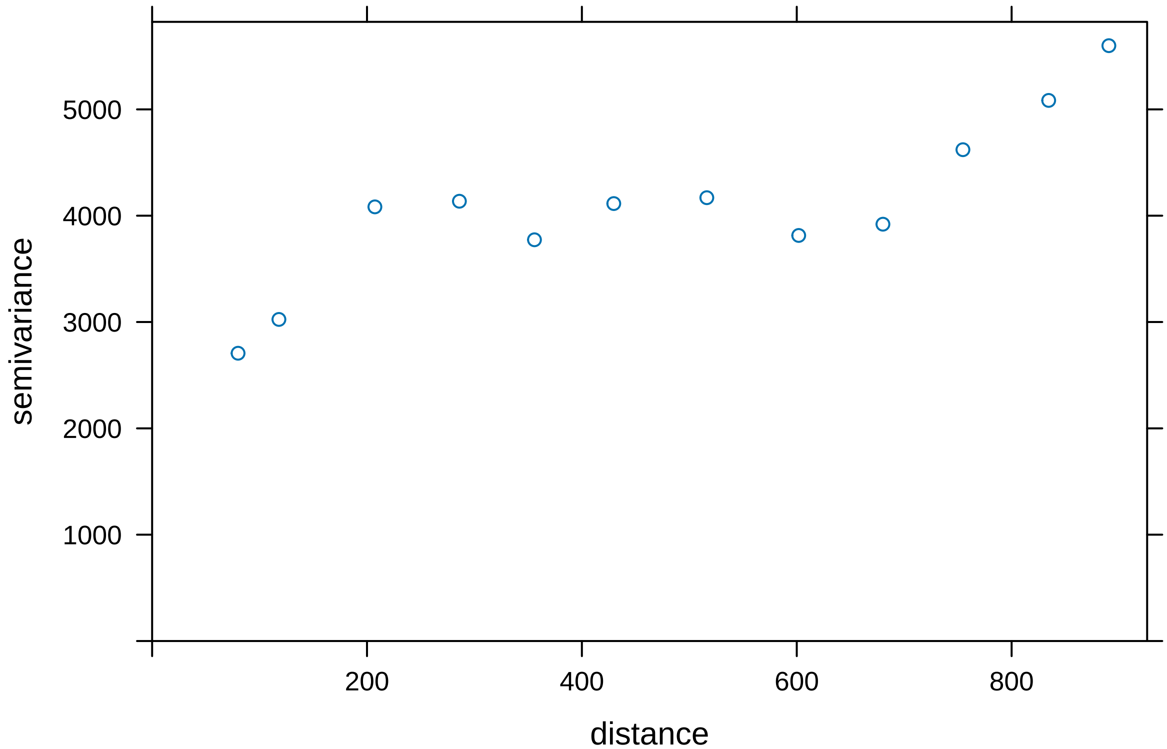
```r
# Build gstat object
# Here is important to change to the variable that is going to be interpolated
g = gstat(formula = Ar ~ CEa+Elev+HSI+NDCI+SBI, data=dados)

# Distance helpers
print(max(dist(dados@coords))/2) #Cutoff - half of the max distance
```

```
## [1] 905.0984
```

```r
print(min(dist(dados@coords))) #Min distance to define the lags (width)
```

```
## [1] 79.991
```

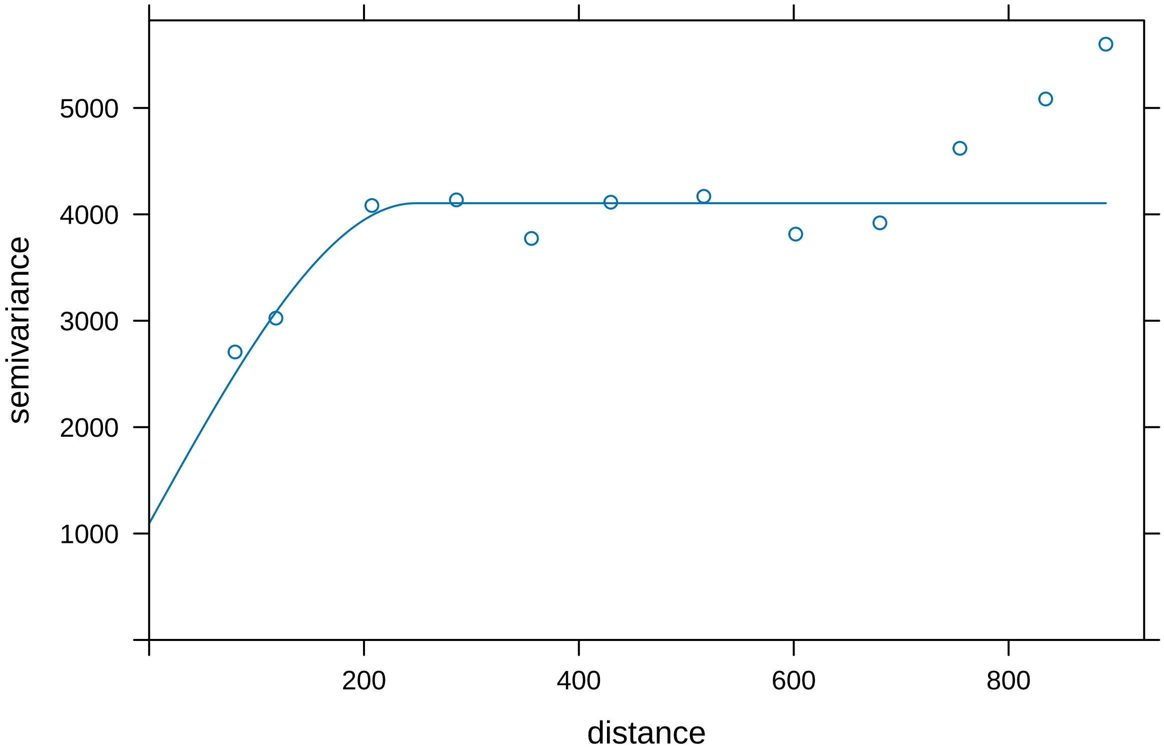
```r
# Experimental semivariogram
var_exp = gstat::variogram(g, cutoff=905.09, width=80, cressie=F)
plot(var_exp)
```




```
var(dados$Ar) # Data variance (should be close to variogram sill)
```

```
## [1] 12874.14
```

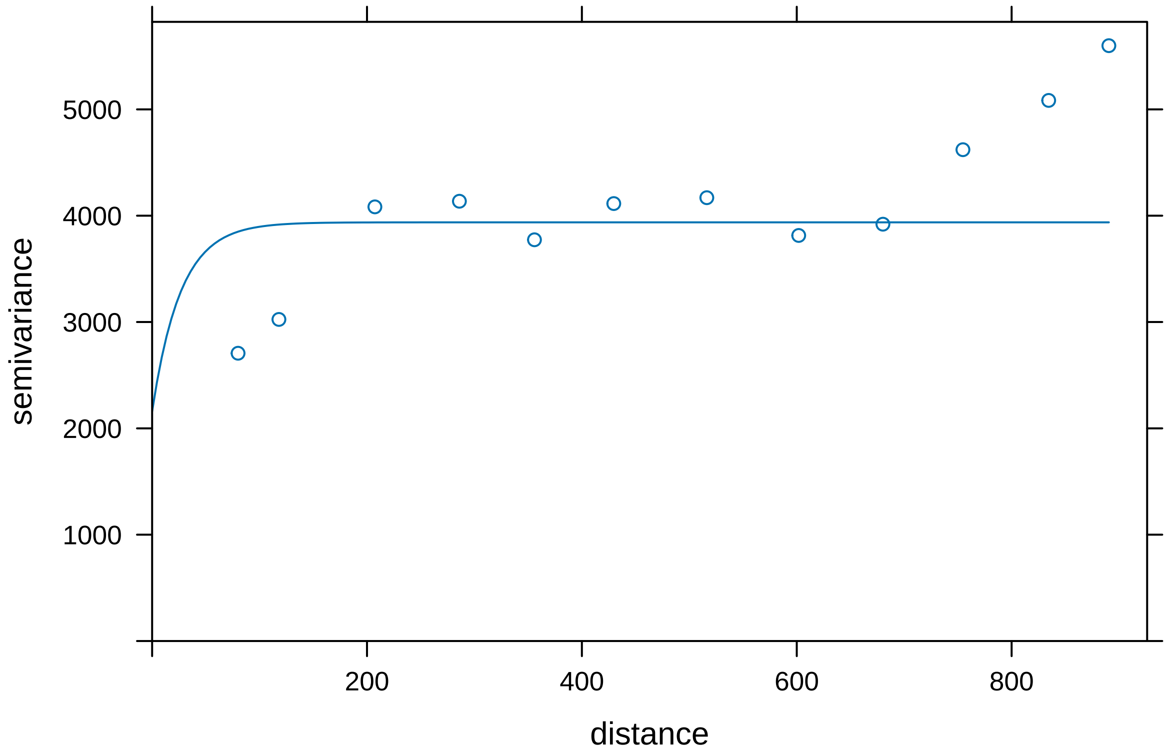
```
#sph
fit.sph = fit.variogram(var_exp, vgm(2000, "Sph", 200, 2000))
plot(var_exp, fit.sph)
```




```r
#exp
fit.exp = fit.variogram(var_exp, vgm(2000, "Exp", 200, 2000))
```

```
## Warning in fit.variogram(var_exp, vgm(2000, "Exp", 200, 2000)): No convergence
## after 200 iterations: try different initial values?
```

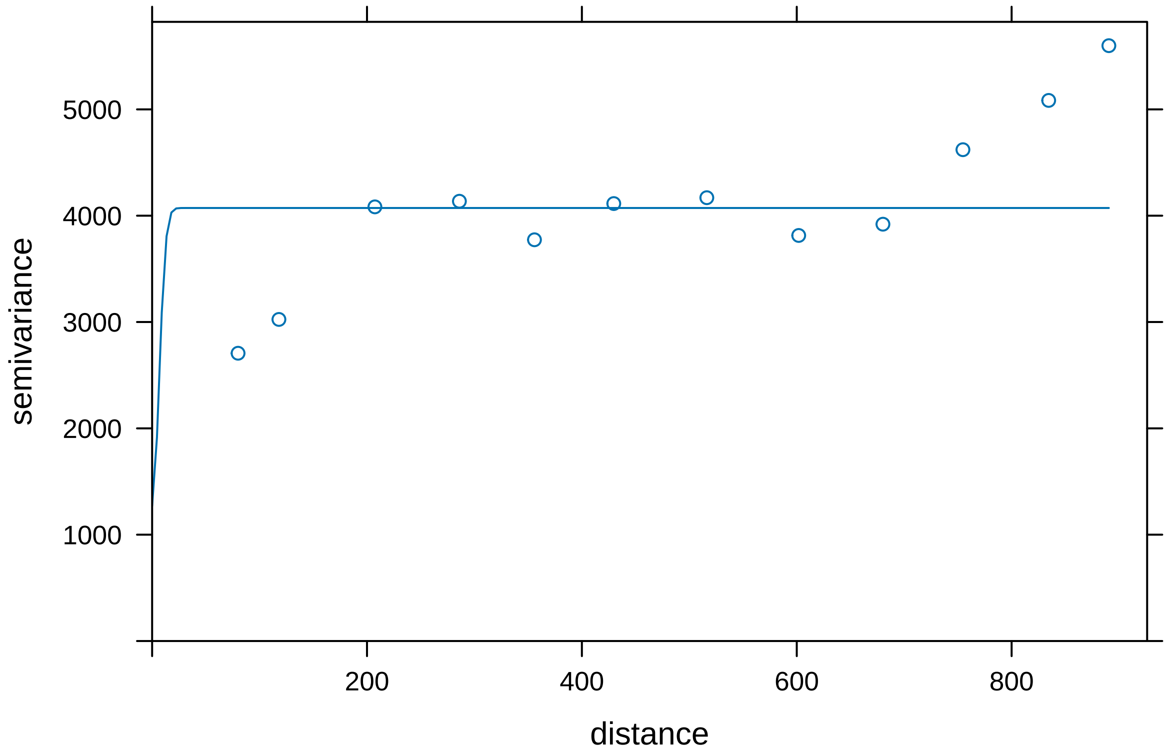
```r
plot(var_exp, fit.exp)
```




```r
#gau
fit.gauss = fit.variogram(var_exp, vgm(2500, "Gau", 200,1500))
```

```
## Warning in fit.variogram(var_exp, vgm(2500, "Gau", 200, 1500)): singular model
## in variogram fit
```

```r
plot(var_exp, fit.gauss)
```



##LOOCV

Cross validation to select the best fit model to krige

```
# Spherical
xvalid.sph <- gstat::krige.cv(Ar ~ CEa+Elev+HSI+NDCI+SBI, locations = dados, model = fit.sph)
lm_sph     <- lm(xvalid.sph$var1.pred ~ xvalid.sph$observed)
r2_sph     <- summary(lm_sph)$r.squared
rmse_sph   <- hydroGOF::rmse(xvalid.sph$var1.pred, xvalid.sph$observed)
slope_sph  <- lm_sph$coefficients[2]
ccc_sph    <- as.numeric(epiR::epi.ccc(xvalid.sph$var1.pred, xvalid.sph$observed)$rho.c["est"])

# Exponential
xvalid.exp <- gstat::krige.cv(Ar ~ CEa+Elev+HSI+NDCI+SBI, locations = dados, model = fit.exp)
lm_exp     <- lm(xvalid.exp$var1.pred ~ xvalid.exp$observed)
r2_exp     <- summary(lm_exp)$r.squared
rmse_exp   <- hydroGOF::rmse(xvalid.exp$var1.pred, xvalid.exp$observed)
slope_exp  <- lm_exp$coefficients[2]
ccc_exp    <- as.numeric(epiR::epi.ccc(xvalid.exp$var1.pred, xvalid.exp$observed)$rho.c["est"])

# Gaussian
xvalid.gau <- gstat::krige.cv(Ar ~ CEa+Elev+HSI+NDCI+SBI, locations = dados, model = fit.gauss)
lm_gau     <- lm(xvalid.gau$var1.pred ~ xvalid.gau$observed)
r2_gau     <- summary(lm_gau)$r.squared
rmse_gau   <- hydroGOF::rmse(xvalid.gau$var1.pred, xvalid.gau$observed)
slope_gau  <- lm_gau$coefficients[2]
ccc_gau    <- as.numeric(epiR::epi.ccc(xvalid.gau$var1.pred, xvalid.gau$observed)$rho.c["est"])
```

Creating the metrics table

```
df.r2    <- data.frame(r2_sph, r2_exp, r2_gau)
df.rmse  <- data.frame(rmse_sph, rmse_exp, rmse_gau)
df.slope <- data.frame(slope_sph, slope_exp, slope_gau)
df.ccc   <- data.frame(ccc_sph, ccc_exp, ccc_gau)


temp <- data.frame(cbind(t(df.r2), t(df.rmse), t(df.slope), t(df.ccc)))
colnames(temp) <- c("R2", "RMSE", "Slope", "LCCC")

rnames <- gsub("r2_", "", rownames(temp))
rownames(temp) <- rnames

print(temp)
```

```
##            R2     RMSE     Slope      LCCC
## sph 0.7749087 53.61169 0.7663367 0.8718756
## exp 0.6422110 67.61898 0.6602810 0.7865822
## gau 0.6314784 68.65663 0.6547472 0.7799837
```

Taking theorical parameters - it has to change for the best model

```
# Extract nugget, partial sill, total sill, and range
nugget        <- fit.sph$psill[1]
partial_sill  <- fit.sph$psill[2]
sill          <- partial_sill + nugget
range         <- fit.sph$range[2]

# Spatial Dependence Index (SDI) as percentage of partial sill over total sill
SDI <- (partial_sill / sill) * 100

# Classify SDI into categories
Class <- ifelse(SDI < 20, "Very low",
         ifelse(SDI < 40, "Low",
         ifelse(SDI < 60, "Medium",
         ifelse(SDI < 80, "High", "Very high"))))

# Build summary table (note: mixing numeric and character will coerce to character)
table_params <- data.frame(
  Parameter = c("Nugget", "Sill", "Range", "SDI%", "Class"),
  Value     = c(nugget, sill, range, SDI, Class),
  stringsAsFactors = FALSE
)

table_params
```

```
##   Parameter            Value
## 1    Nugget 1093.70655939007
## 2      Sill 4104.45654193808
## 3     Range 247.796862976812
## 4      SDI% 73.3531943092853
## 5     Class             High
```

##Krige

Doing the kriging

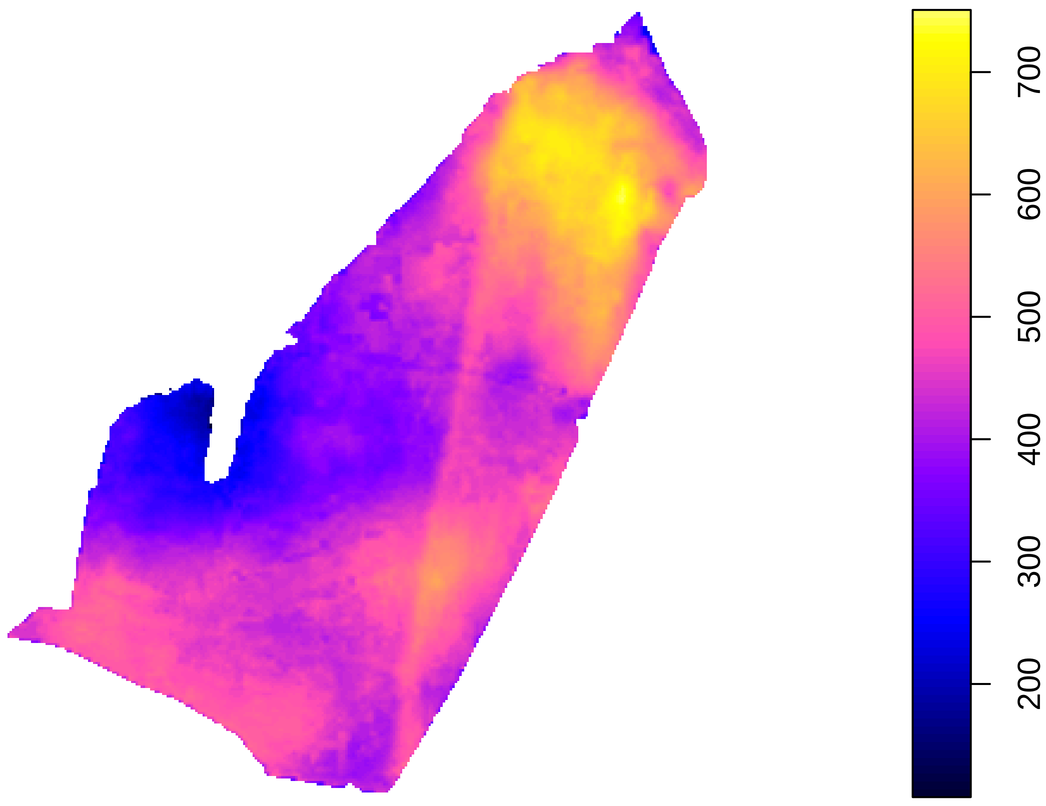
```r
# Kriging with the selected model
proj4string(dados) <- CRS("+init=epsg:32723")
proj4string(dados)
```

```
## [1] "+proj=utm +zone=23 +south +datum=WGS84 +units=m +no_defs"
```

```r
mapa = krige(Ar ~ CEa+Elev+HSI+NDCI+SBI, locations = dados, newdata=grid,model = fit.sph)
```

```
## [using universal kriging]
```

```r
plot(mapa)
```



Exporting the raster

```r
mapaRaster <- raster(mapa)

filename<-"G:/Mi unidad/02_Maestria/01_Projeito/13_GitHub/Ar_KED_1cada1_Paulinia.tiff"
writeRaster(mapaRaster, filename , format = 'GTiff', overwrite = T)
```

Graphic with ggplot

```r
ggplot() +
  geom_raster(data = as.data.frame(mapa), aes(fill = var1.pred, x = x, y = y)) +
  scale_fill_paletteer_c("ggthemes::Red-Gold") +
  labs(fill = expression('Argila (g*kg'^-1*')'))+
  annotation_north_arrow(which_north = "grid",height = unit(1, "cm"),
                         width = unit(0.9, "cm"),
                         pad_x = unit(0.5, "cm"),
                         pad_y = unit(5, "cm"),
                         style=north_arrow_fancy_orienteering())+
  annotation_scale(location = "bl", width_hint = 0.2)+
  theme_bw()
```

```
## Using plotunit = 'm'
```