M1683. Intensificación en Saberes Técnicos: Diseño y Gestión de Bases de Datos Territoriales. Geoestadística II

Domingo Rasilla

02 enero, 2022

Contents

PROYECCIONES EN gstat (LLEVAR A GEOSTRADISTICA 1 YA QUE FALLA EL EJERCICIO DE LAS TEM OREGON) **2**

1. LECTURA DE LOS DATOS **5**

2. KRIGING CON EXTERNAL DRIFT **8**

3. KRIGNING CON REGRESIÓN **13**

4. INDICATOR KRIGING **17**

4.1 Umbrales 17

4.2 Categorical variables 23

5. SIMULACIÓN GEOESTADÍSTICA **29**

5.1 Simulación gaussiana 30

5.2 Simulación de indicador 33

6. EJERCICIOS **35**

Código para añadir los shapefiles 36

Código para diseñar una rejilla para predicciones 37

En esta sesión continuaremos con el análisis geoestadístico, utilizando los mismos datos de la sesión anterior y los mismos paquetes

```
pkgs <- c("gstat",
          "stars",
          "RColorBrewer",
          "sf")

install.packages(pkgs)
```

```
library(ggplot2)
library(gstat)
library(RColorBrewer)
library(sf)
library(stars)
library(viridis)
```

PROYECCIONES EN gstat (LLEVAR A GEOSTRADISTICA 1 YA QUE FALLA EL EJERCICIO DE LAS TEM OREGON)

De forma predeterminada, **gstat** asume que los datos siguen una proyección cartesiana, a menos que el objeto Spatial* que contiene los datos tenga metadatos de proyección que especifiquen que está en un sistema de coordenadas esférico (lat / lon).

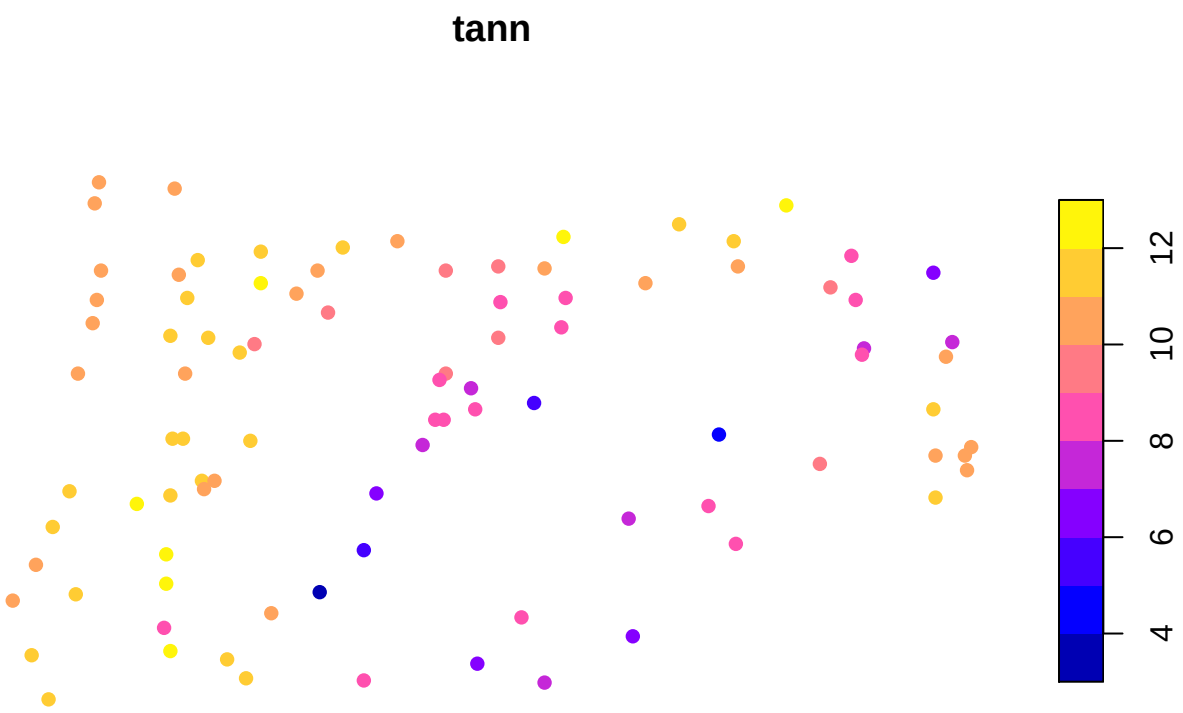
Sin estos metadatos, las coordenadas de longitud y latitud se tratarán como cartesianas y las distancias se calcularán incorrectamente para el análisis de variogramas y kriging. Podemos ilustrar esta circunstancia a partir de los datos de temperatura media del estado de Oregón. Deberá cargarse primero y preguntar al programa por su proyección (esto debería leer 'NA'):

```
oregon <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/oregon/oregontann.shp", quiet = TRUE)

st_crs(oregon)
```

```
## Coordinate Reference System: NA
```

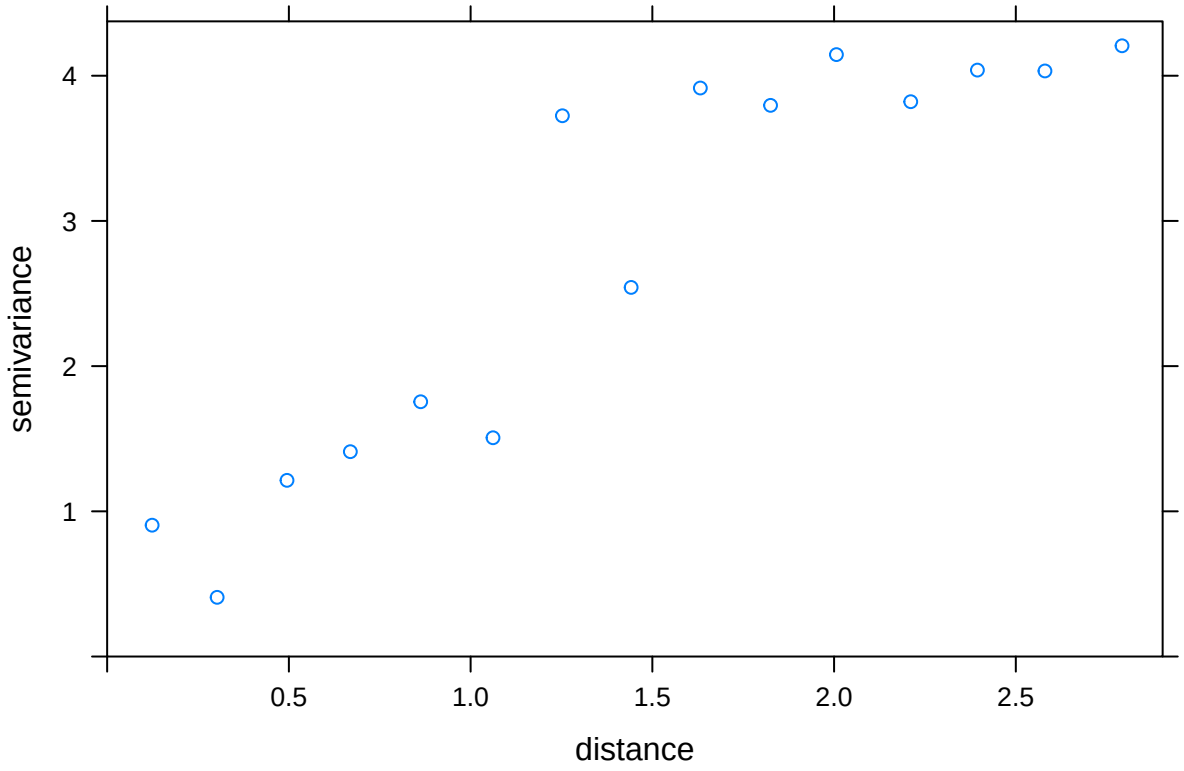
Podemos cartografiar rápidamente esos datos

```
# names(oregon)

plot(oregon["tann"], pch = 16)
```



La figura muestra un patrón espacial bastante evidente, con temperaturas más cálidas a la izquierda (elevaciones más bajas y más cerca de la costa). Ahora haremos un variograma para explorar este patrón (debe estar cargado el paquete **gstat**):

```
or_vgm <- variogram(tann ~ 1, oregon)

plot(or_vgm)
```



Los desfases en distancias aparecen calculados en grados, de acuerdo con los comentarios anteriores.

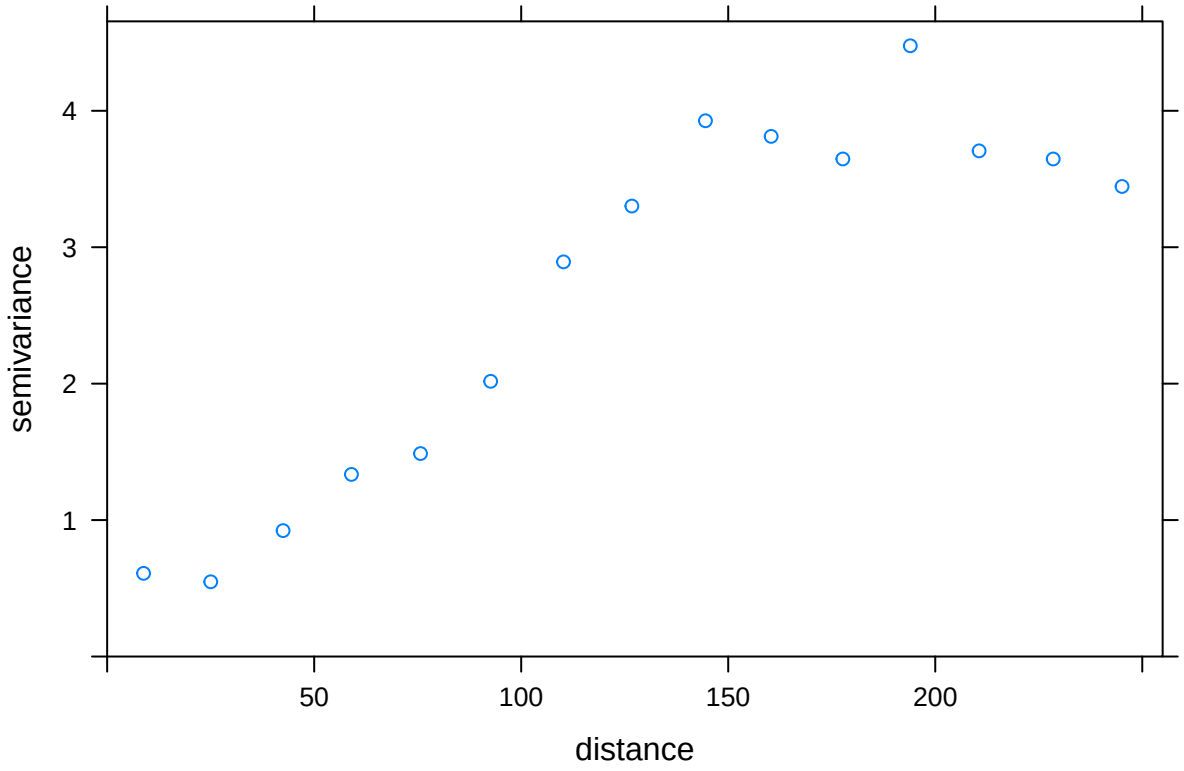
Para corregir esta circunstancia, podemos proyectar los datos o simplemente especificar que estos datos son coordenadas esféricas. En general, la segunda opción es mejor, a menos que se esté trabajando en un área muy pequeña, ya que todas las proyecciones distorsionarán las distancias y/o direcciones sobre áreas grandes.

En consecuencia, establecemos el `st_crs` en [EPSG 4326] epsgID, que es el estándar WGS84:

```
st_crs(oregon) <- 4326
```

Si volvemos a realizar el variograma, **gstat** ya reconoce las coordenadas como coordenadas esféricas y calcula distancias como distancias de gran círculo en km, en lugar de grados euclidianos, reduciendo el sesgo en las distancias.

```
or_vgm <- variogram(tann ~ 1, oregon)

plot(or_vgm)
```



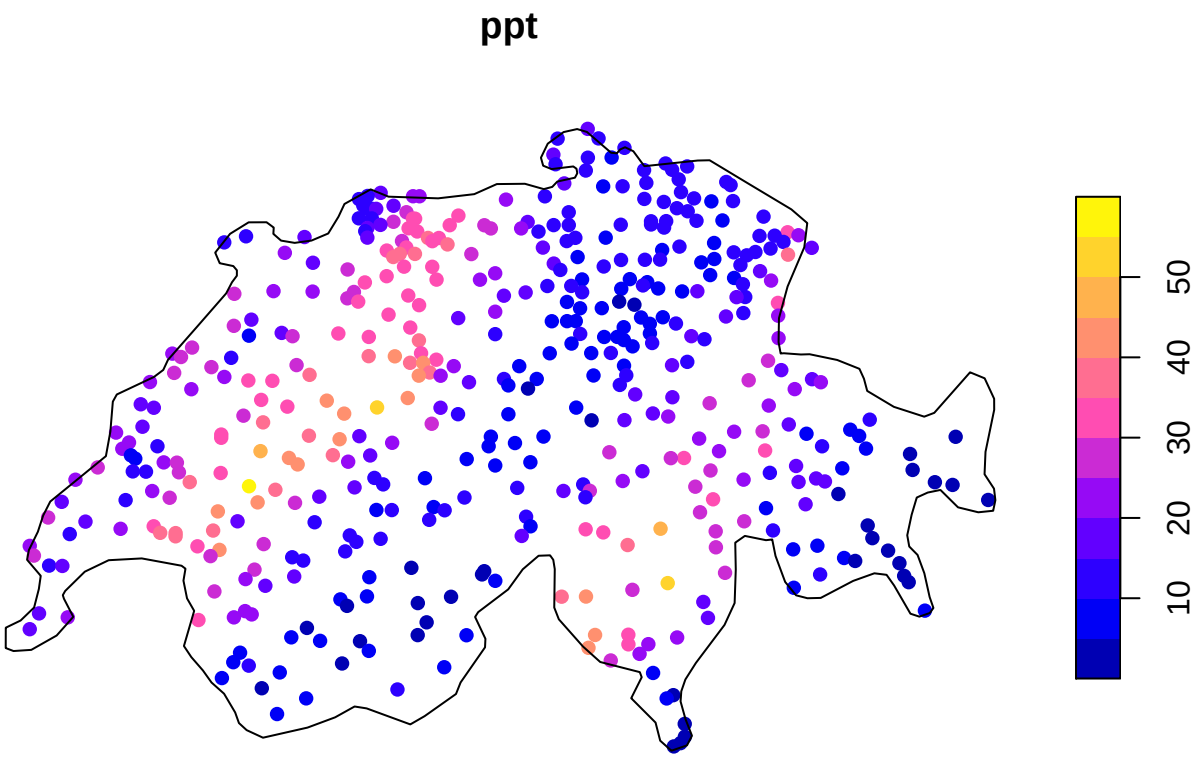
1. LECTURA DE LOS DATOS

Este código es una copia del elaborado en la sesión anterior; cargará y convertirá todos los archivos necesarios para trabajar con los datos de precipitación en Suiza.

```
## Precipitation data
swiss <- read.csv("D:/Docencia_Master_2021/GEOG6000/sesion05/swiss_ppt/swiss_ppt.csv")
swiss.sf <- st_as_sf(swiss,
                     coords = c("x", "y"),
                     crs = 2056)

## Elevation grid
swiss.dem <- read_stars("D:/Docencia_Master_2021/GEOG6000/sesion05/swiss_ppt/swiss_dem.asc")
st_crs(swiss.dem) <- 2056

## Swiss border
countries <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/ne_50m_admin_0_countries/ne_50m_admin_0
swiss.bord <- subset(countries, NAME == "Switzerland")
swiss.bord <- st_transform(swiss.bord, 2056)
```

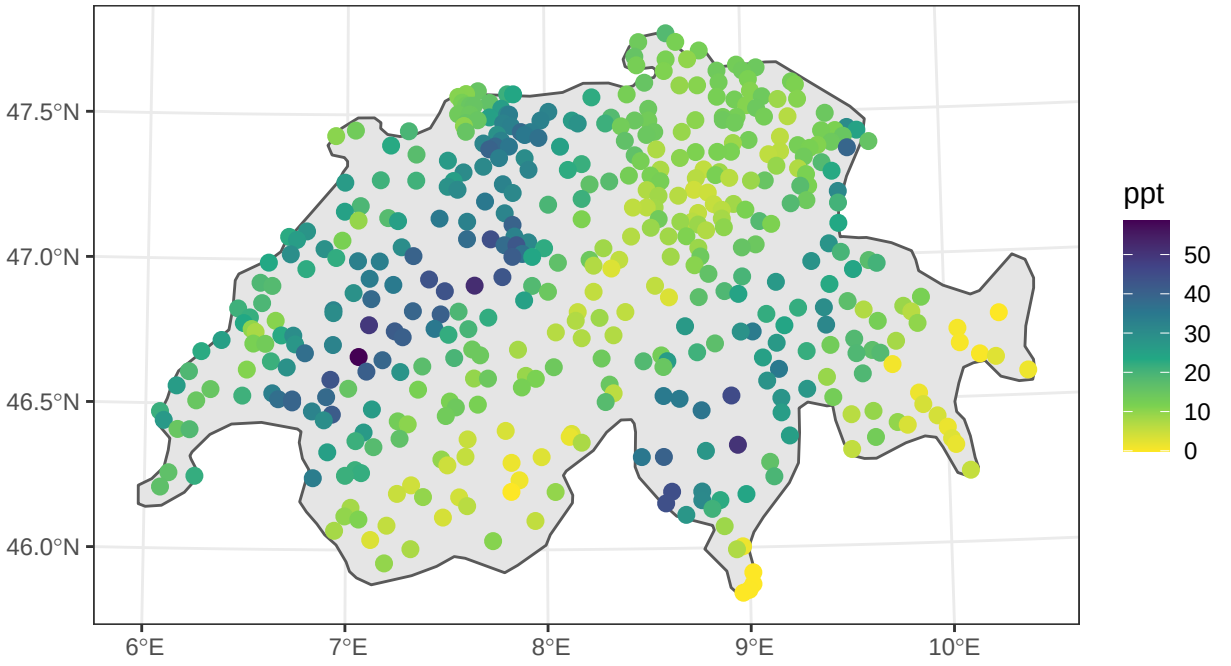
En primer lugar, merece la pena realizar un simple gráfico con las cantidades de precipitación

```
plot(swiss.sf["ppt"], reset = FALSE, pch = 16)
plot(st_geometry(swiss.bord), add = TRUE)
```



Ese mismo gráfico se puede mejorar sustancialmente con la librería **ggplot2**:

```
ggplot() +
  geom_sf(data = swiss.bord) +
  geom_sf(data = swiss.sf, aes(col = ppt), size = 2.5) +
  scale_color_viridis_c(direction = -1) +
  theme_bw()
```

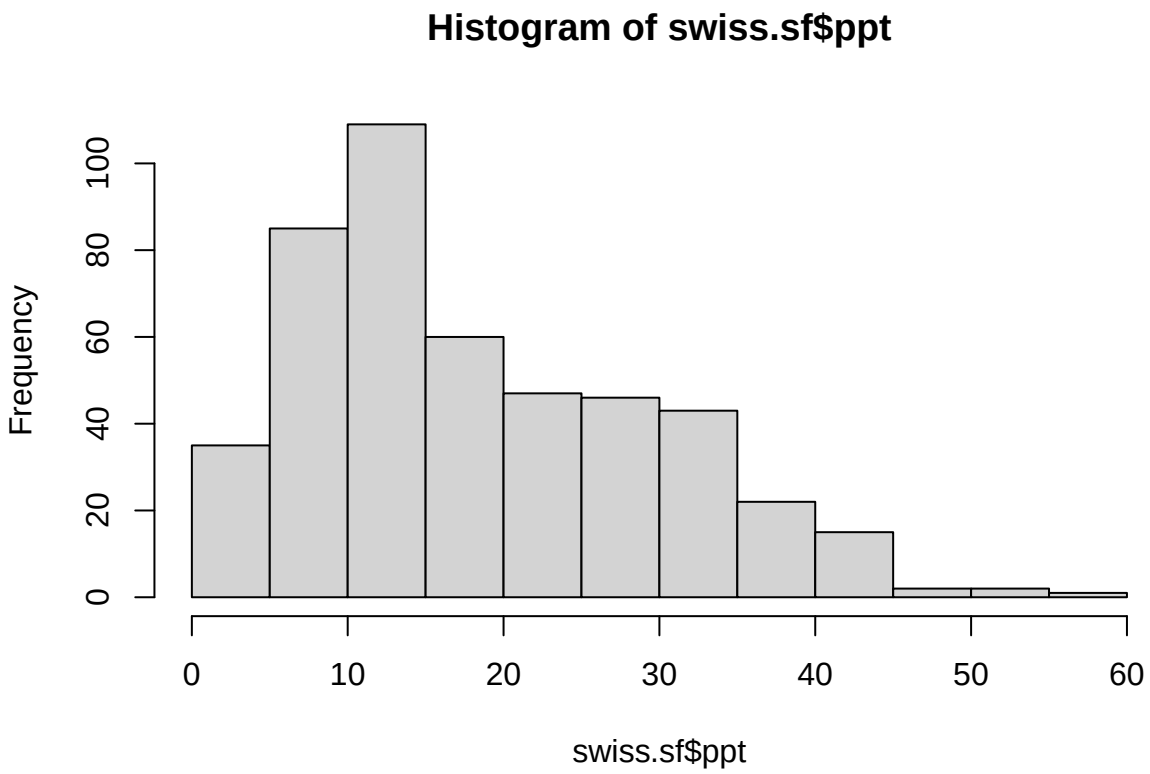


A continuación se debe cargar la base de datos correspondiente al río Mosa:

```
meuse <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/meuse/meuse.shp", quiet = TRUE)
st_crs(meuse) <- 28992

meuse.riv <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/meuse/meuseriv.shp", quiet = TRUE)
st_crs(meuse.riv) <- 28992

meuse.grid <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/meuse/meusegrid.shp", quiet = TRUE)
st_crs(meuse.grid) <- 28992
meuse.grid <- st_rasterize(meuse.grid["dist"], dx = 40, dy = 40)
```

Un histograma de los datos de precipitación muestra que están sesgados a la derecha. Como en ocasiones anteriores, los transformaremos calculando su logarítmo para normalizar la distribución. Además, dado que varias estaciones poseen ceros, también agregamos un pequeño valor positivo para hacer posible la transformación logarítmica.

```
hist(swiss.sf$ppt)
```



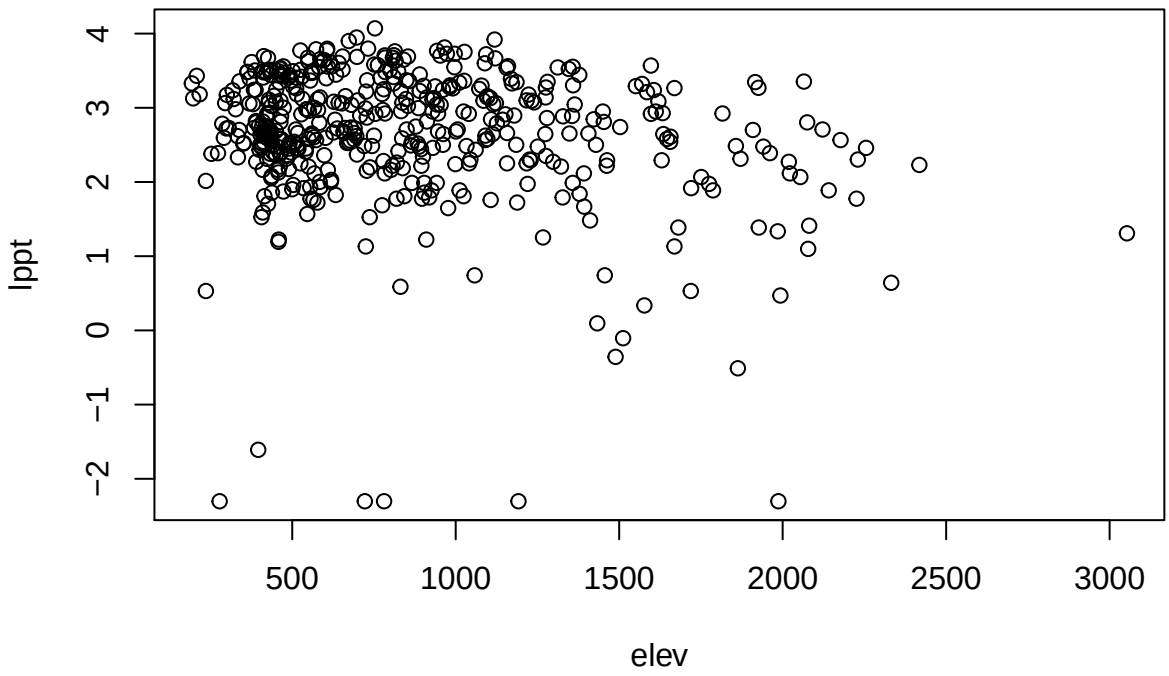

```
swiss.sf$lppt <- log(swiss.sf$ppt + 1e-1)
```

2. KRIGING CON EXTERNAL DRIFT

Kriging puede incluir covariables para mejorar la predicción. Uno de los métodos más flexibles es el kriging con una deriva externa, donde la deriva se refiere a cualquier covariable que se haya registrado tanto en los lugares de muestreo como en los lugares de predicción.

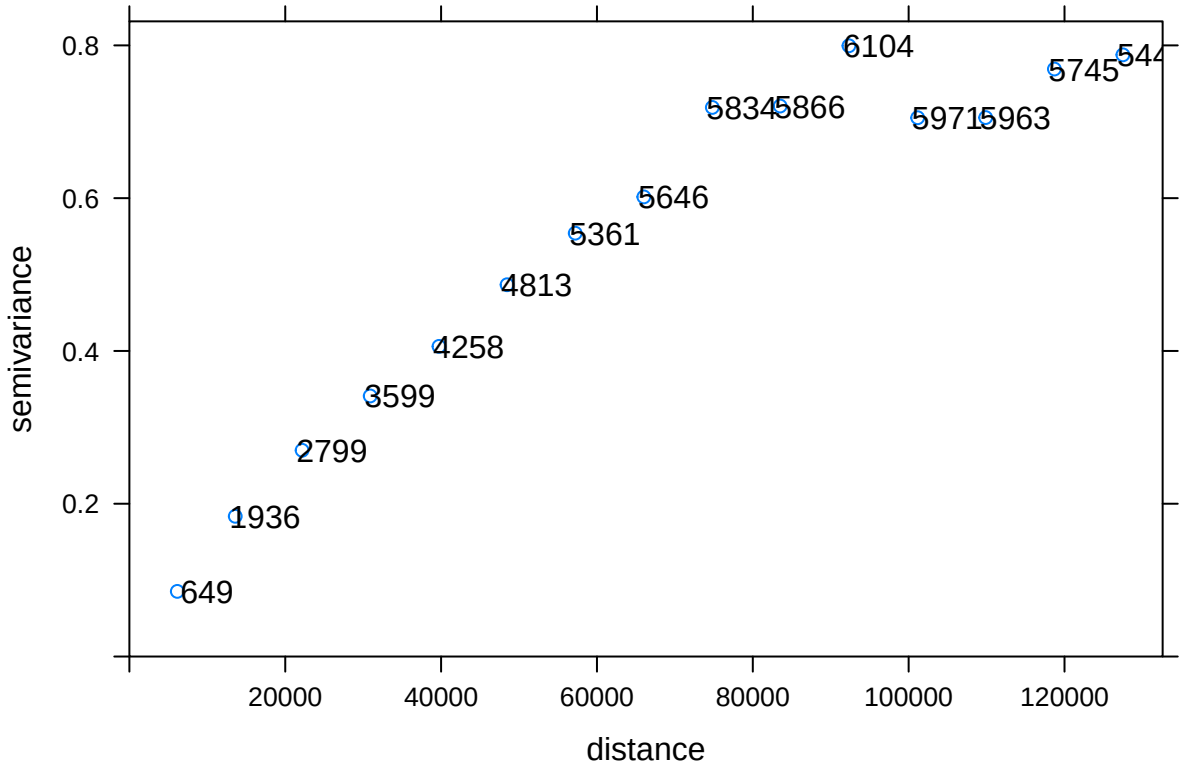
Un diagrama de dispersión de la precipitación contra la elevación de cada sitio muestra una relación negativa débil, con valores generalmente más altos en elevaciones más bajas:

```
plot(lppt ~ elev, swiss.sf)
```



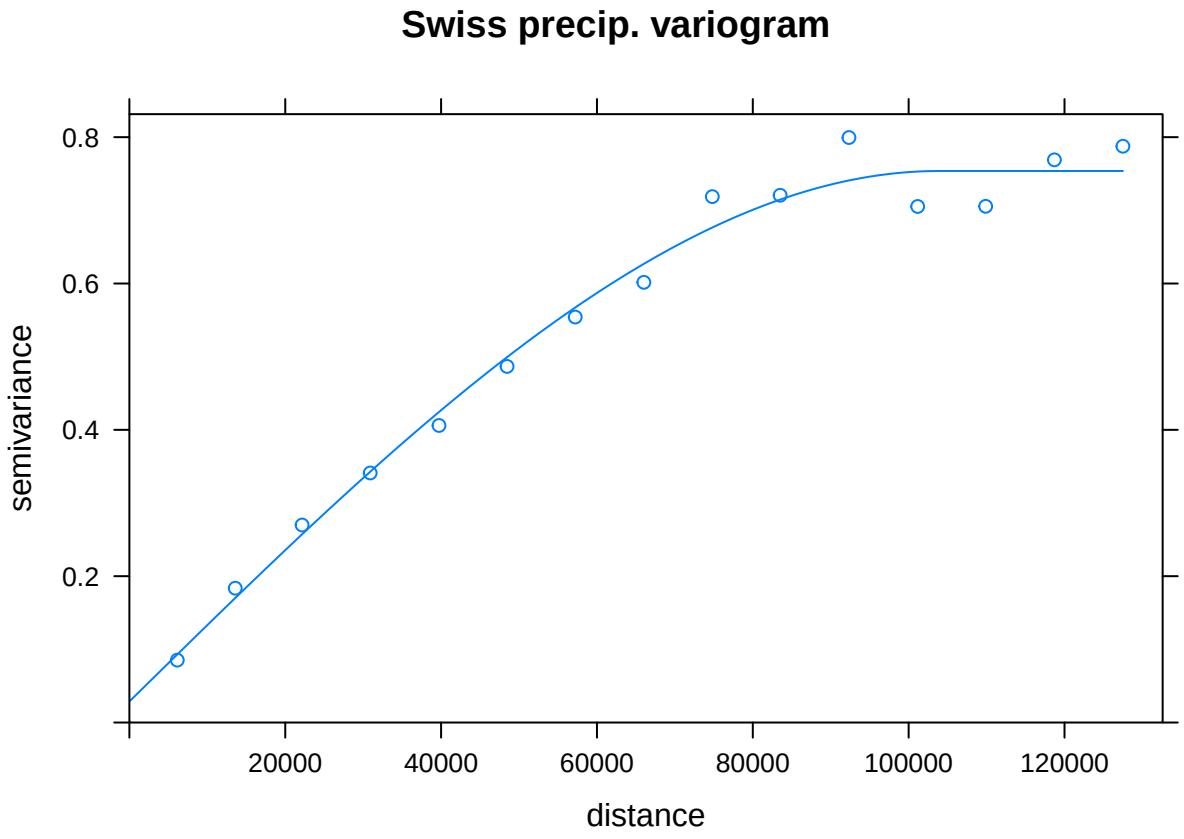
Usaremos este modelo junto con los datos de elevación del DEM para realizar kriging con una deriva externa en los datos de precipitación. Como estamos incorporando una covariable en nuestro modelo, primero necesitamos rehacer el variograma de precipitación, para que el variograma tenga esto en cuenta:

```
ppt.var <- variogram(lppt ~ elev, swiss.sf)

plot(ppt.var, plot.numbers = TRUE)
```



A continuación ajustamos el modelo:

```
modNugget <- 0.05
modRange <- 100000
modSill <- 0.75

ppt.vgm1 <- vgm(psill = modSill,
                "Sph",
                range = modRange,
                nugget = modNugget)

ppt.vgm2 <- fit.variogram(ppt.var, ppt.vgm1)

plot(ppt.var, ppt.vgm2, main = "Swiss precip. variogram")
```



A continuación, este variograma es usado para interpolar los datos de precipitación con la misma función que en la práctica de laboratorio anterior (`krige ()`), pero ahora especificando `elev` como variable independiente en la fórmula del modelo. Esto requiere que tanto los puntos espaciales (`swiss.sf`) como las nuevas ubicaciones (`swiss.dem`) tengan una variable llamada 'elev ', así que primero deber verificarse esta circunstancia:

```
## Check to see that both the data and grid have elev variables
names(swiss.sf)
```

```
## [1] "id"       "ppt"      "elev"     "geometry" "lppt"
```

```
names(swiss.dem)
```

```
## [1] "swiss_dem.asc"
```

Al solicitar los nombres de las variables, detectamos que la variable `swiss.dem` fue etiquetada usando el nombre del fichero original. Para cambiarlo procedemos de la siguiente manera:

```
names(swiss.dem) <- "elev"
```

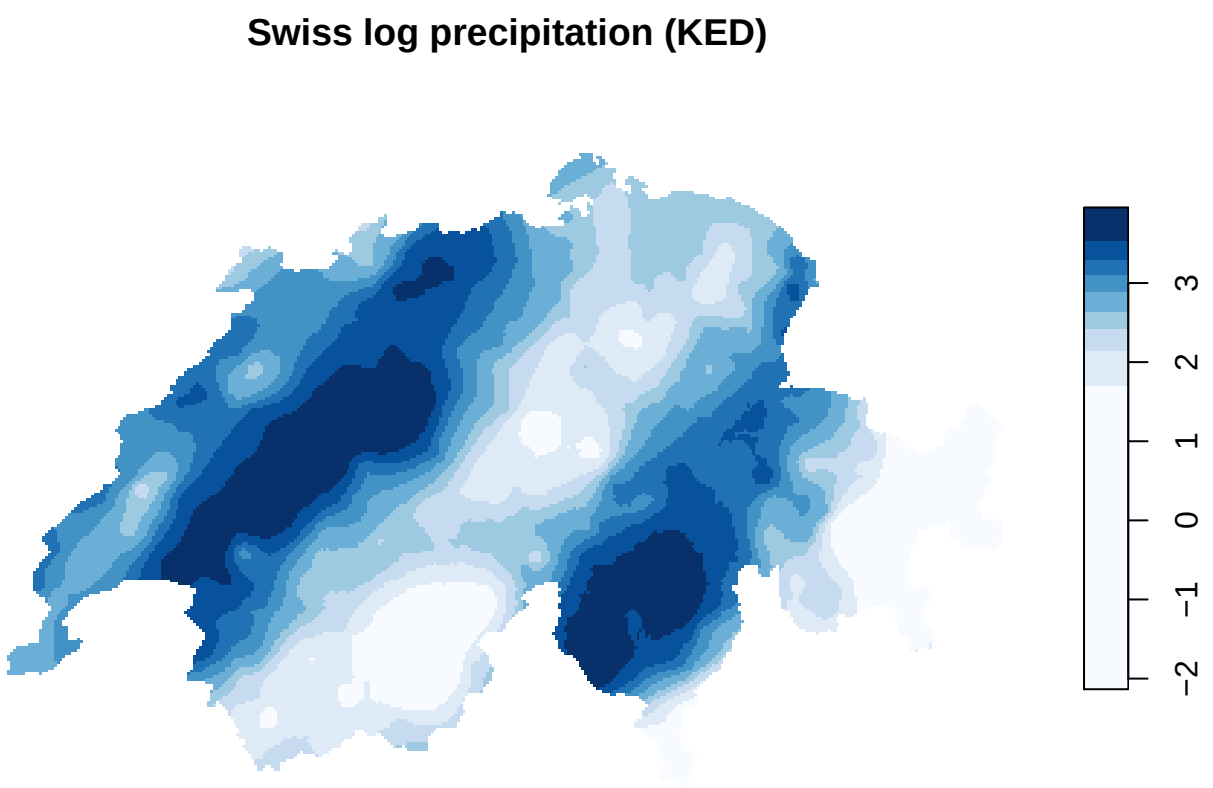
Ahora mismo ya se puede continuar creando el modelo:

```
ppt.pred.ked <- krige(lppt ~ elev,                    ## kriging with external drift
                     swiss.sf,
                     swiss.dem,
                     ppt.vgm2)
```

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

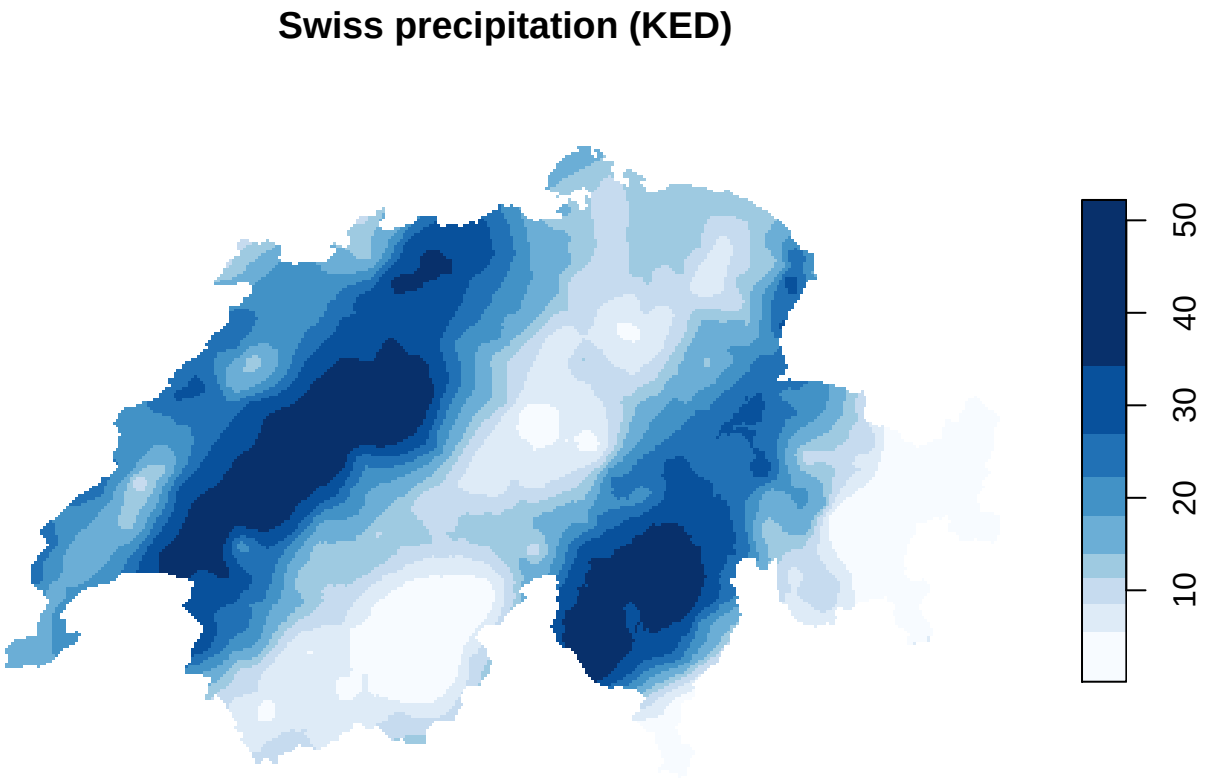
Con los resultados procedemos a cartografiar las nuevas predicciones:

```
# names(ppt.pred.ked)

my.pal = brewer.pal(9, "Blues")
plot(ppt.pred.ked["var1.pred"],
     col = my.pal,
     main = "Swiss log precipitation (KED)")
```



Como señalamos, el modelo está construido a partir de los datos transformados en logarítmos. Para obtener el modelo con valores similares a los originales volvemos a trasformar la precipitación en mm usando la función`exp()` y cartografiándola:

```
ppt.pred.ked$ppt <- exp(ppt.pred.ked$var1.pred)

plot(ppt.pred.ked["ppt"],
     col = my.pal,
     main = "Swiss precipitation (KED)")
```



Como en casos anteriores, es posible estimar la calidad de las predicciones del modelo usando validación cruzada

```
ppt.cv.ked <- krige.cv(lppt ~ elev,
                       swiss.sf,
                       ppt.vgm2,
                       nmax = 40,
                       nfold = 5)
```

```
## RMSE
sqrt(mean(ppt.cv.ked$residual^2))
```

```
## [1] 0.4114167
```

```
## R2
cor(ppt.cv.ked$observed, ppt.cv.ked$var1.pred)^2
```

```
## [1] 0.7960134
```

3. KRIGNING CON REGRESIÓN

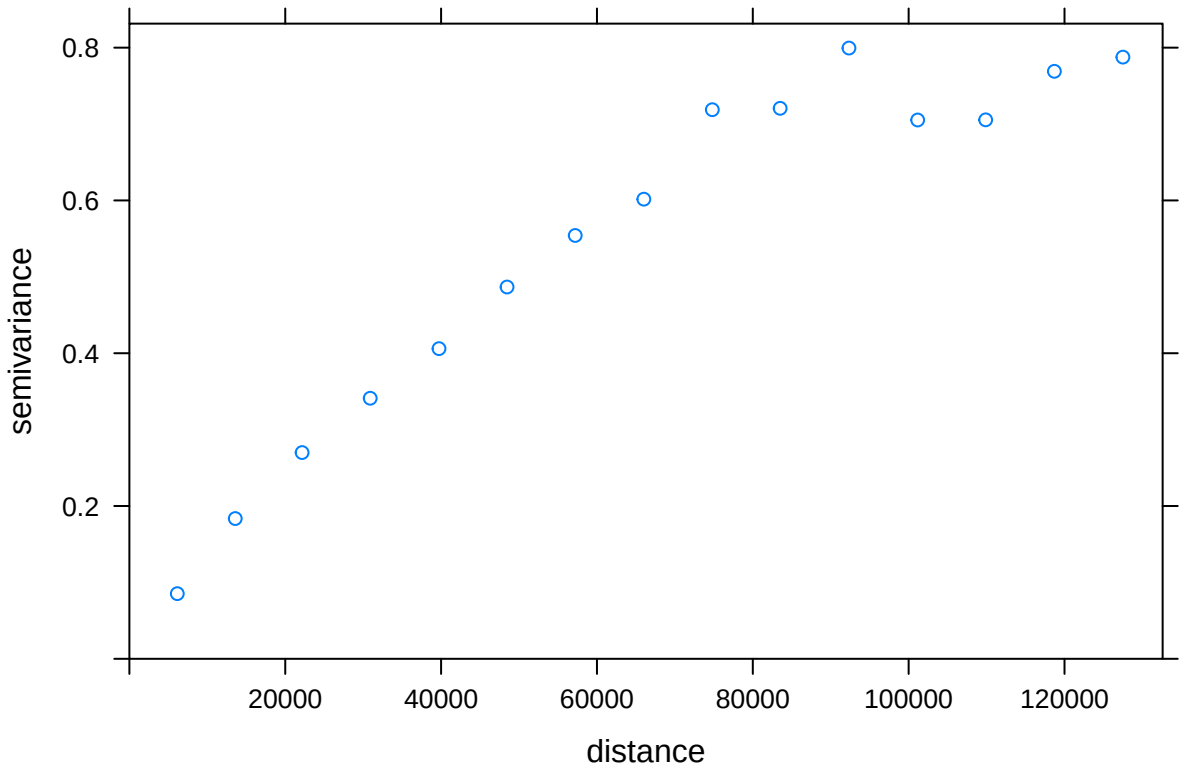
El kriging con regresión proporciona un enfoque más flexible para incluir covariables que el kriging universal o el kriging de deriva externa, pero requiere un poco más de trabajo. La idea es que en lugar de incorporar

una relación potencialmente complicada en el modelo, esto se modele por separado, luego los residuos se interpolan usando kriging simple.

Luego, se puede hacer una estimación final en cada nueva ubicación agregando el valor predicho del modelo original al residual interpolado. Esto abre la posibilidad de usar cualquier técnica de regresión con la (s) covariable (s) para modelar las tendencias estructurales más grandes, y luego usar kriging simple para modelar las desviaciones de esta tendencia.

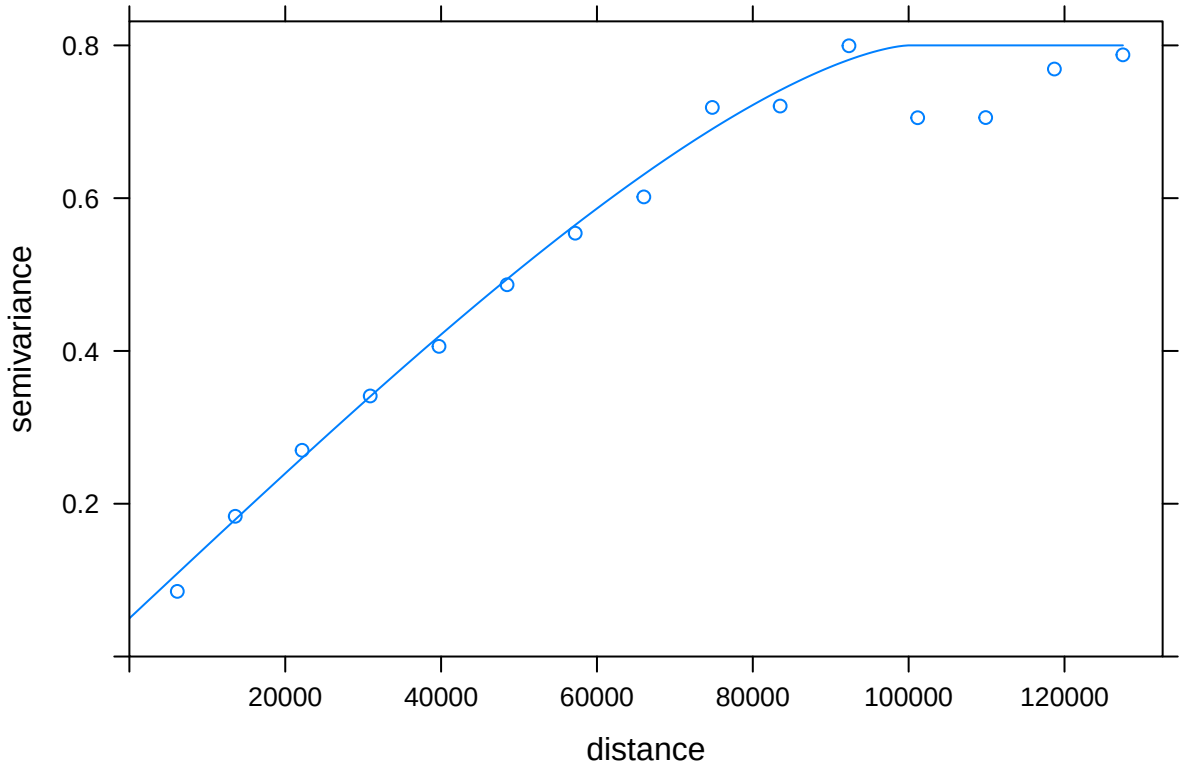
Como ejemplo simple, construiremos un modelo lineal de precipitación usando la elevación como covariable, y usaremos los residuos de esto como base para el kriging. Tenga en cuenta que como hemos especificado la covariable en el modelo de regresión, ya no necesitamos incluirla en el variograma o en la función `krige ()`.

```
fit1 <- lm(lppt ~ elev, swiss.sf)

swiss.sf$resid <- residuals(fit1)

resid.var <- variogram(resid ~ 1, swiss.sf)

plot(resid.var)
```



A continuación se ajusta variograma modelo:

```
resid.vgm <- vgm(0.75, "Cir", 100000, 0.05)

plot(resid.var, resid.vgm)
```



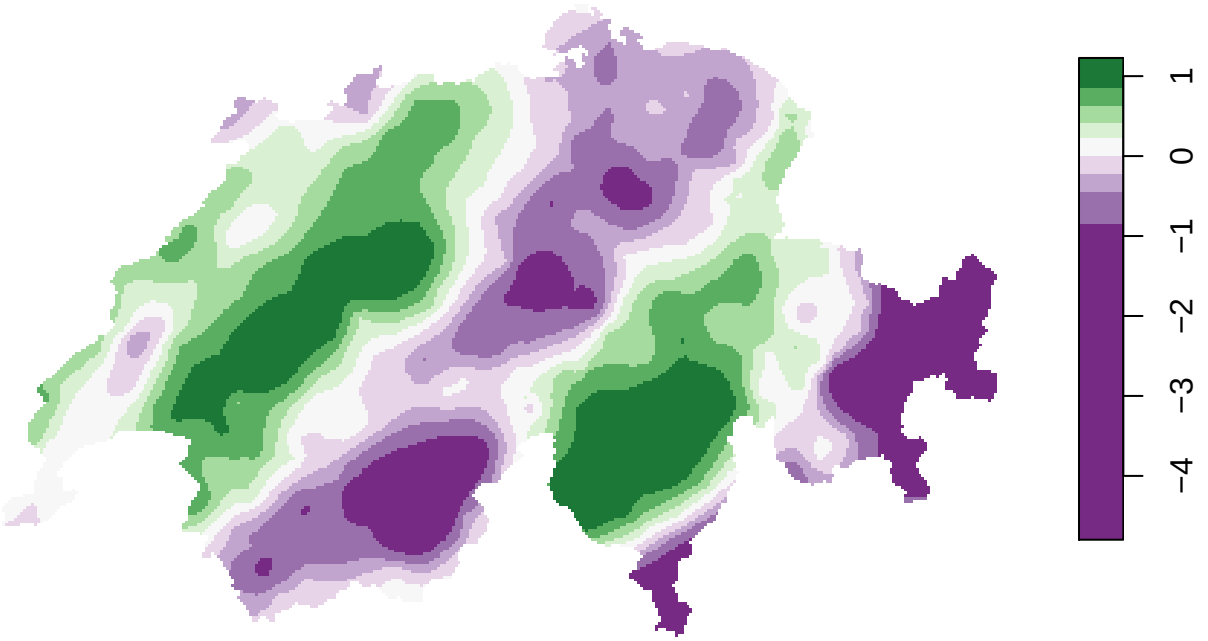
Posteriormente se interpolan los resíduos. Téngase en cuenta que, dado que son resíduos, su valor medio es conocido (es = 0), por lo que se puede usar el kriging simple para la interpolación. El comando incluye el argumento `beta` con un valor de 0 (la media de los resíduos):

```
resid.sk <- krige(resid ~ 1,
                  swiss.sf,
                  swiss.dem,
                  resid.vgm,
                  nmax = 40,
                  beta = 0)
```

```
## [using simple kriging]
```

Cartografiamos los residuos

```
my.pal <- brewer.pal(9, "PRGn")

plot(resid.sk["var1.pred"],
     col = my.pal,
     main = "Swiss precipitation residuals (RK)")
```



Para realizar las estimaciones finales, debe predecirse primero la precipitación usando usando un modelo de regresión simple con la elevación del DEM como variable independiente, y almacenamos los resultados en un objeto `starsDEM`.

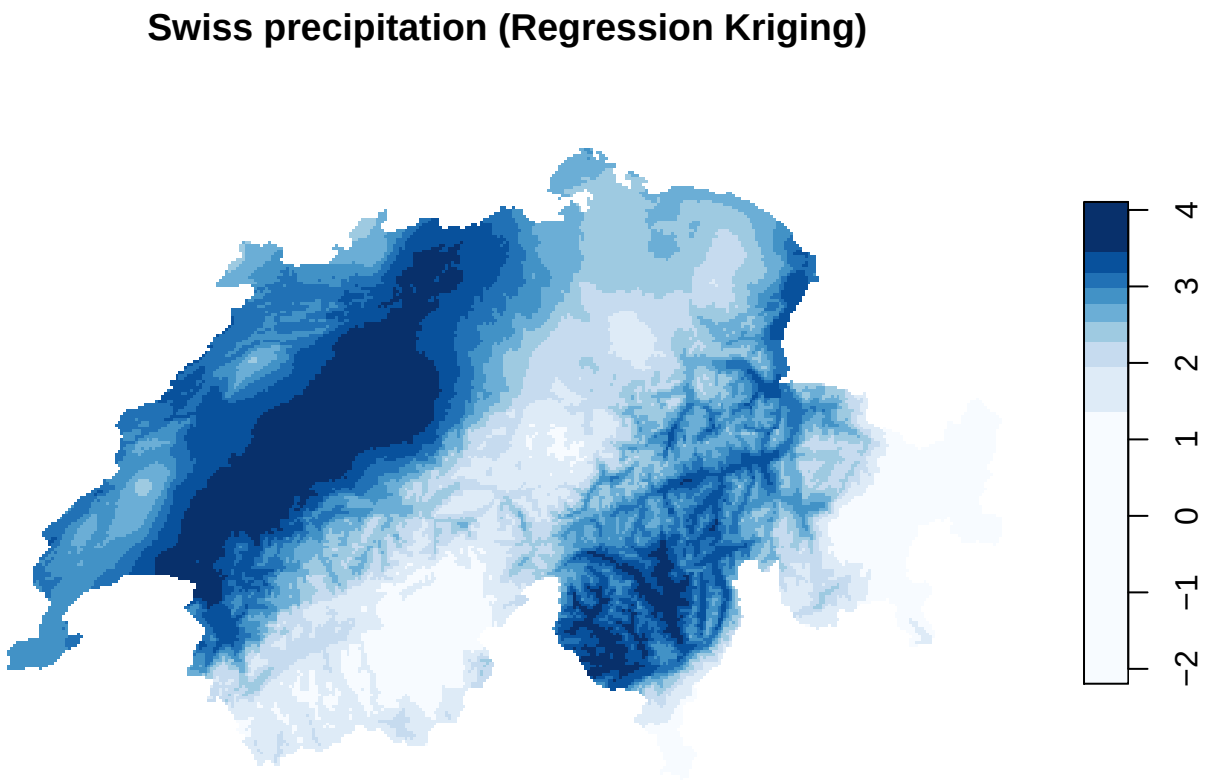
```
swiss.dem$ppt.lm <- predict(fit1, swiss.dem)
```

A continuación se añaden los resíduos interpolados a esos valores:

```
swiss.dem$ppt.rk <- swiss.dem$ppt.lm + resid.sk$var1.pred
```

Y por último se cartografía las predicciones:

```
my.pal <- brewer.pal(9, "Blues")

plot(swiss.dem["ppt.rk"],
     col = my.pal,
     main = "Swiss precipitation (Regression Kriging)")
```



Podemos, por supuesto, volver a transformar los datos a la escala no logarítmica usando `exp ()`. Si bien los resultados aquí no son muy diferentes al kriging con una deriva externa mencionada anteriormente, el método es mucho más flexible. Podríamos reemplazar el modelo lineal con modelos lineales generalizados, modelos aditivos, modelos de efectos mixtos e incluso métodos de aprendizaje automático.

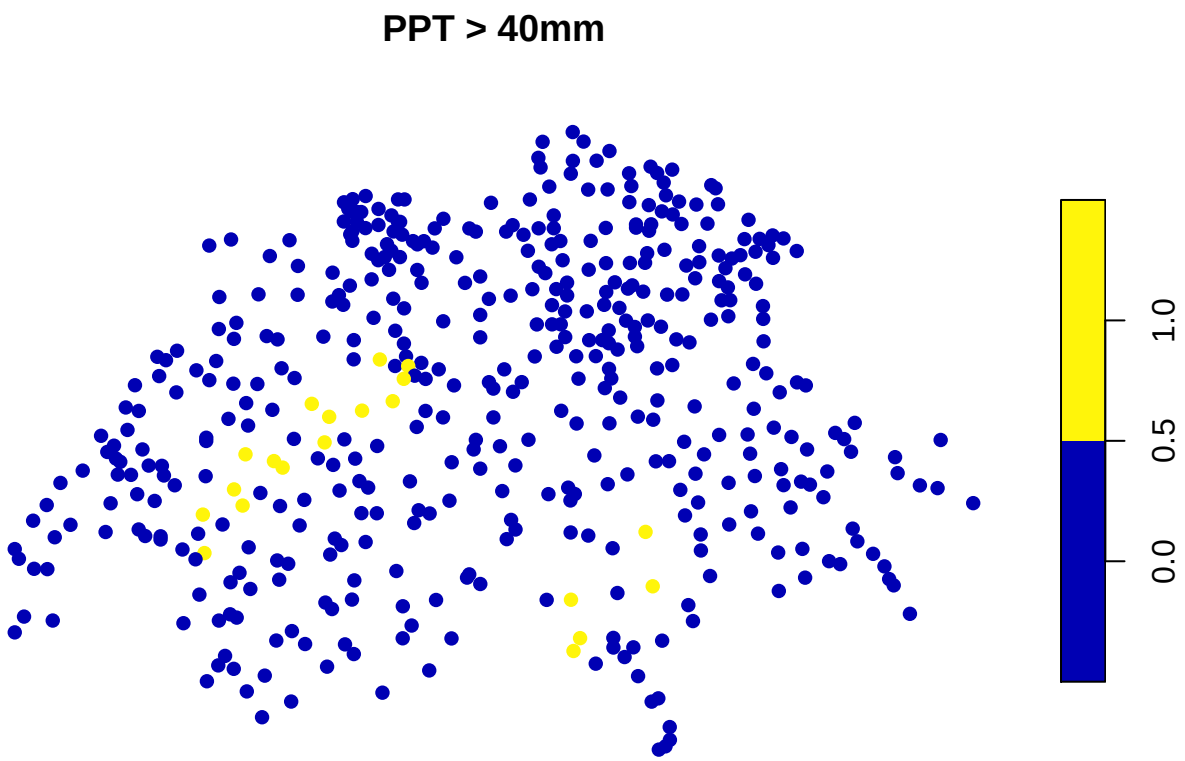
4. INDICATOR KRIGING

Esta modalidad de kriging se utiliza para interpolar variables binarias como probabilidades. Se puede utilizar para estimar si la variable de interés estará por encima o por debajo de un umbral determinado en una nueva ubicación, o la probabilidad de que una nueva ubicación tenga una variable binaria o categórica. En cualquier caso, el método consiste simplemente en interpolar valores binarios (0/1) usando kriging ordinario. Como utiliza la misma función que antes, puede incluir una tendencia o una deriva externa si es necesario.

4.1 Umbrales

Primero usaremos este método para encontrar todas las ubicaciones en Suiza con más de 40 mm de lluvia. Aquí, creamos una nueva variable en el marco de datos `swiss.sf`, que es si la estación tuvo o no $>$ 40 mm de lluvia, y usamos`spplot ()`para hacer una cifra rápida.

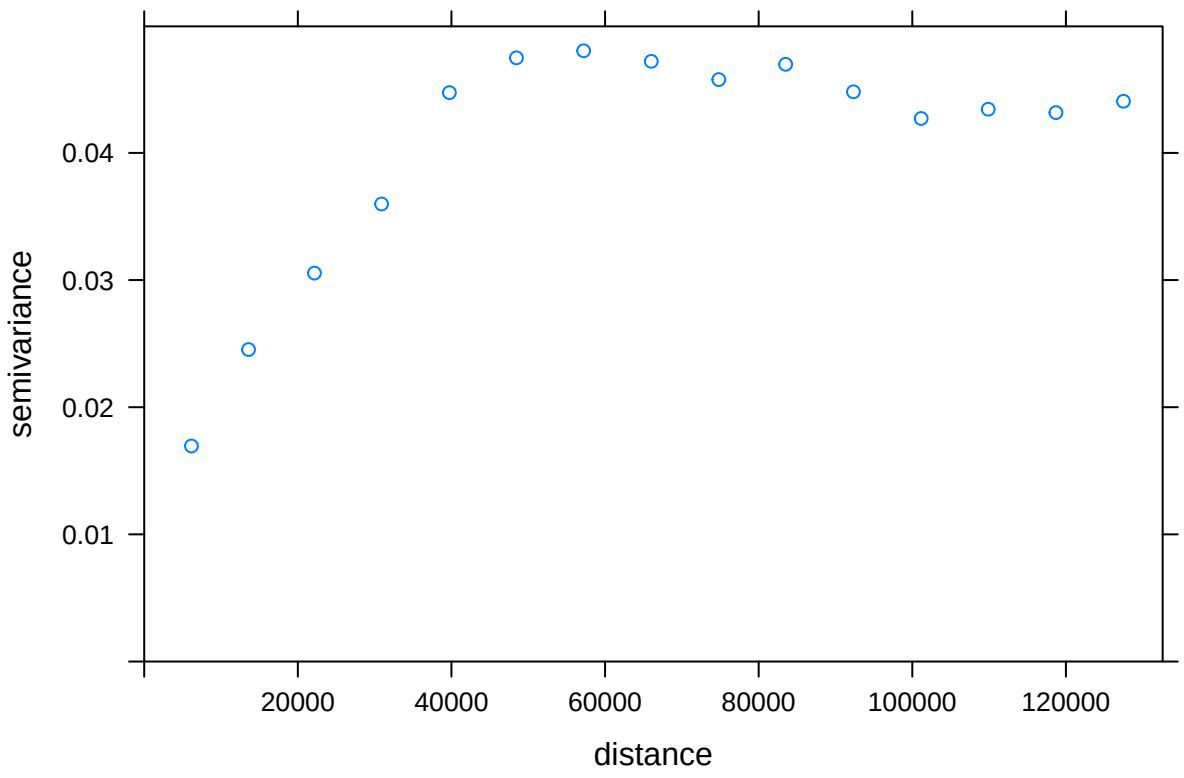
```
swiss.sf$ppt40 <- swiss.sf$ppt > 40

plot(swiss.sf["ppt40"],
     pch = 16,
       main = "PPT > 40mm")
```



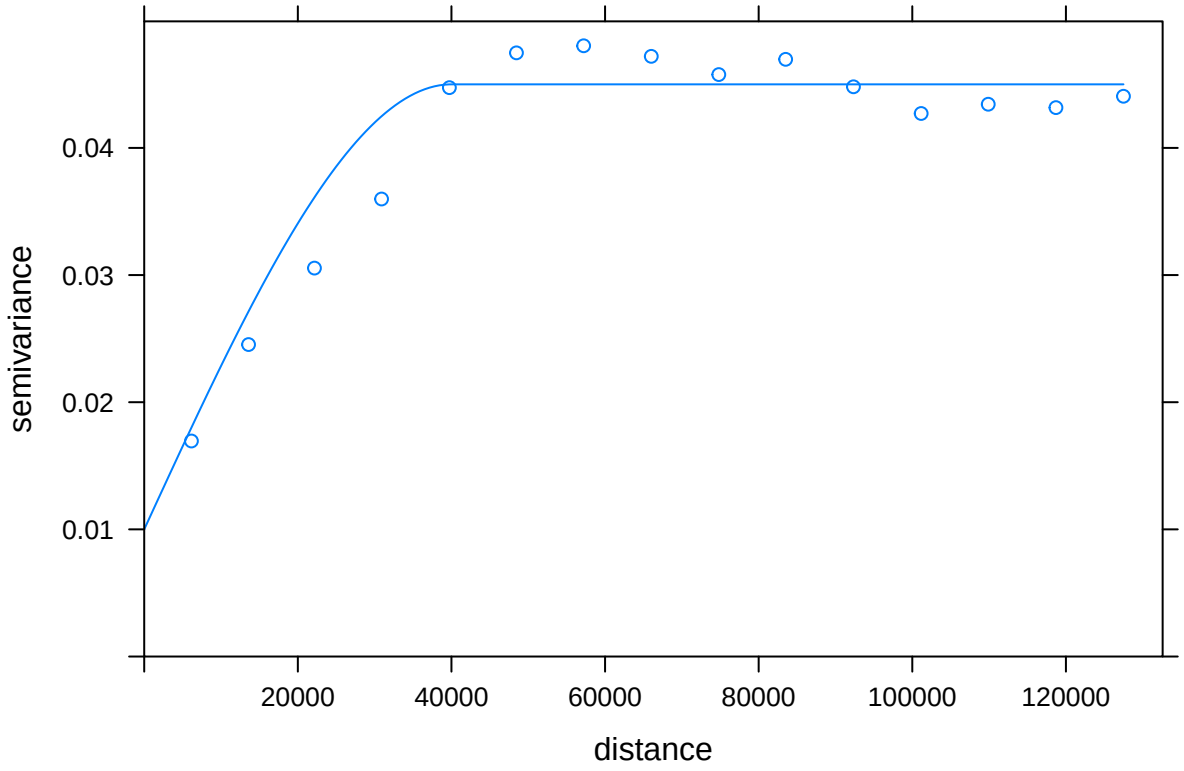
De acuerdo con lo expuesto anteriormente:

1. Se crea el variograma basado en la muestra

```
ppt40.var <- variogram(ppt40 ~ 1, swiss.sf)

plot(ppt40.var)
```

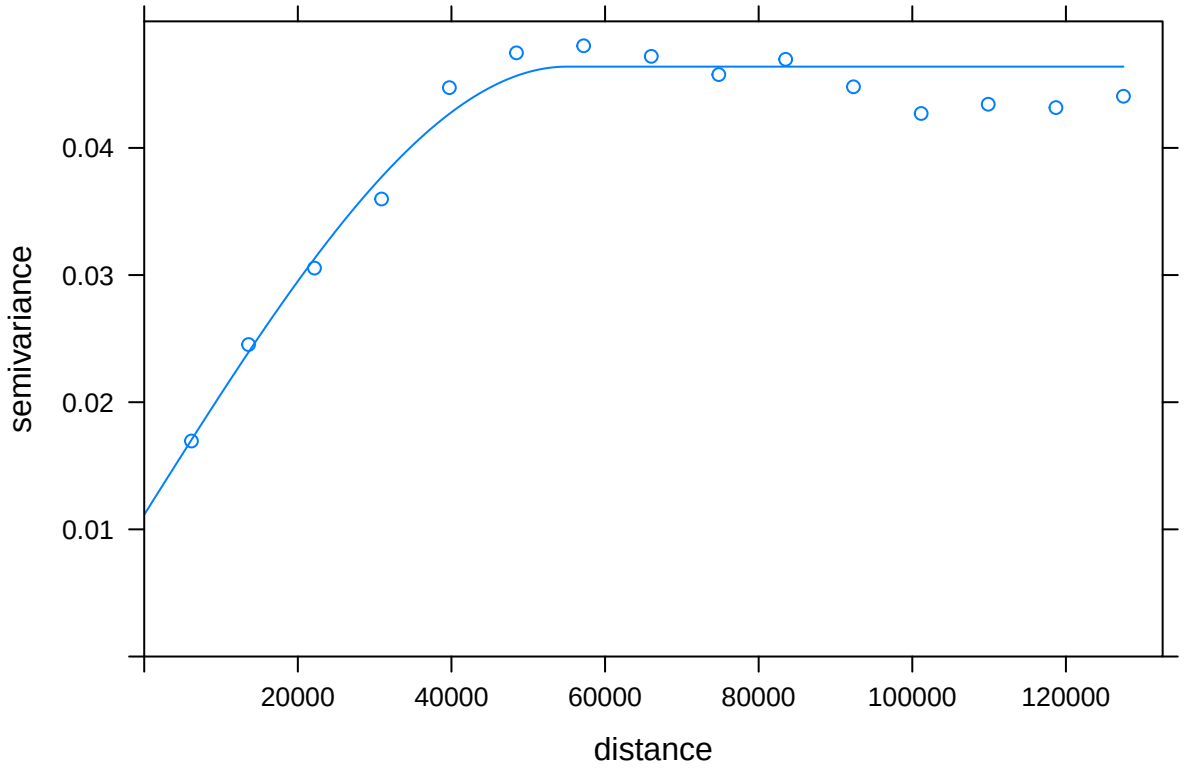



```
ppt40.vgm <- vgm(0.035, "Sph", 40000, 0.01)

plot(ppt40.var, ppt40.vgm)
```



2. Ajuste de un modelo de variograma

```
ppt40.vgm2 <- fit.variogram(ppt40.var, ppt40.vgm)

plot(ppt40.var, ppt40.vgm2)
```



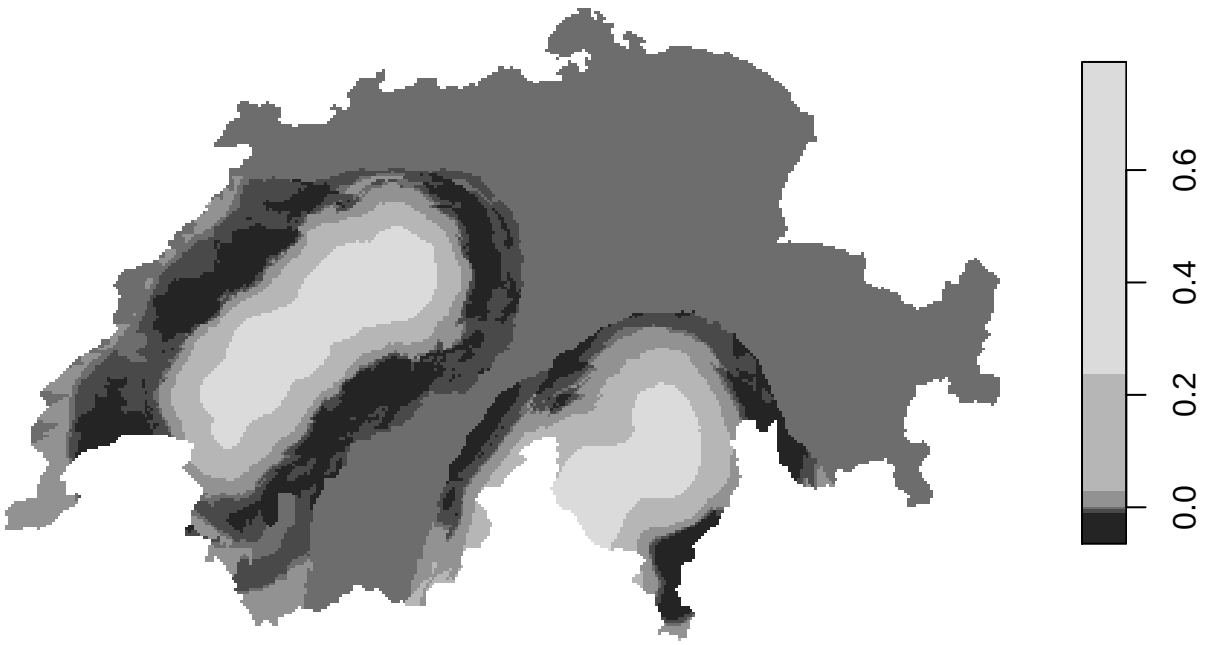
3. Interpolación usando kriging ordinario:

```
ppt40.ik <- krige(ppt40 ~ 1,
                  swiss.sf,
                  swiss.dem,
                  ppt40.vgm2,
                  nmax = 40)
```

```
## [using ordinary kriging]
```

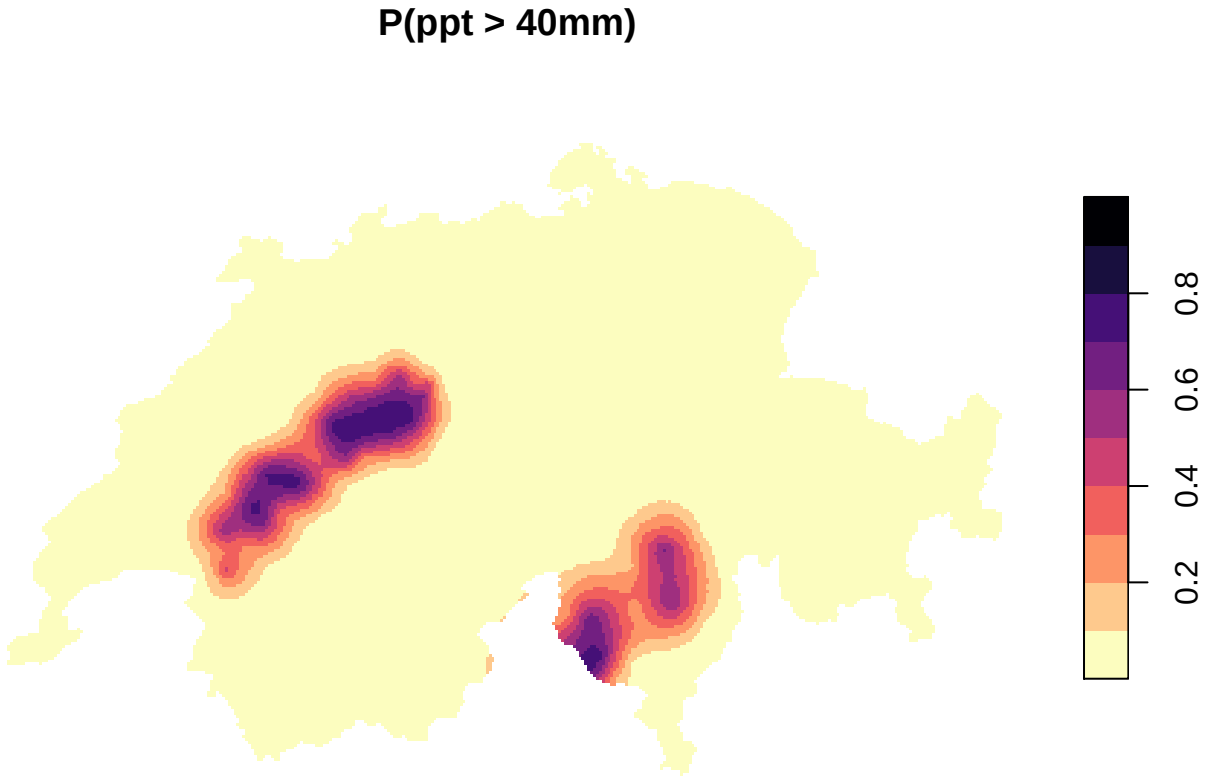
4. Cartografía de las nuevas predicciones:

```
plot(ppt40.ik["var1.pred"])
```



Debe señalarse que las probabilidades obtenidas no son verdaderas (algunos valores son inferiores a < 0). Sin embargo, a los efectos de la interpolación geoestadística, estos se consideran casi como probabilidades y, a menudo, se corrigen entre 0 y 1, que es la siguiente tarea. Para ello, primerso se usa la función `which ()` para encontrar todos los píxeles con un valor por debajo de cero, asignándoles un valor cero. Luego trazamos usando una de las paletas de colores **viridis**:

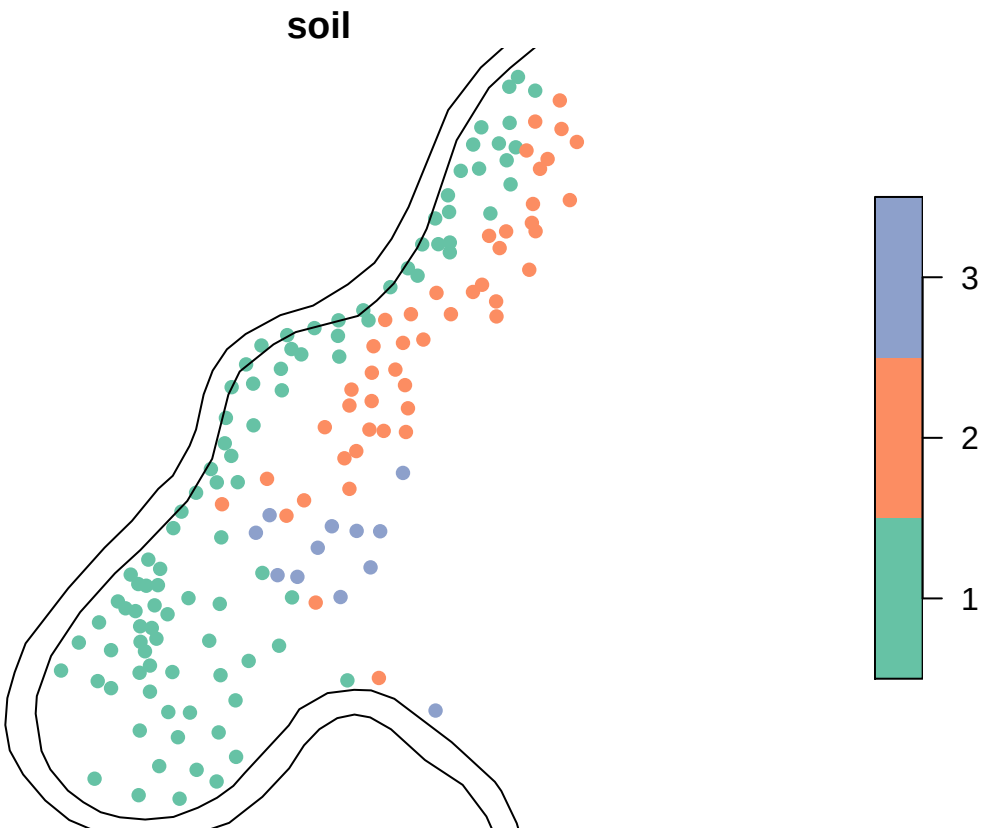
```
ppt40.ik$var1.pred[which(ppt40.ik$var1.pred < 0)] <- 0

my.pal <- rev(viridis::magma(10))

plot(ppt40.ik["var1.pred"],
     col = my.pal,
     breaks = seq(0, 1, length.out = 11),
     main = "P(ppt > 40mm)")
```



4.2 Categorical variables

El procedimiento explicado en líneas anteriores también se puede utilizar para variables categóricas. Por ejemplo, el conjunto de datos Mosa contiene una variable que representa el tipo de suelo para cada uno de los sitios de muestreo, dividido en tres clases. La función `spplot ()` nos permite ver su distribución espacial:

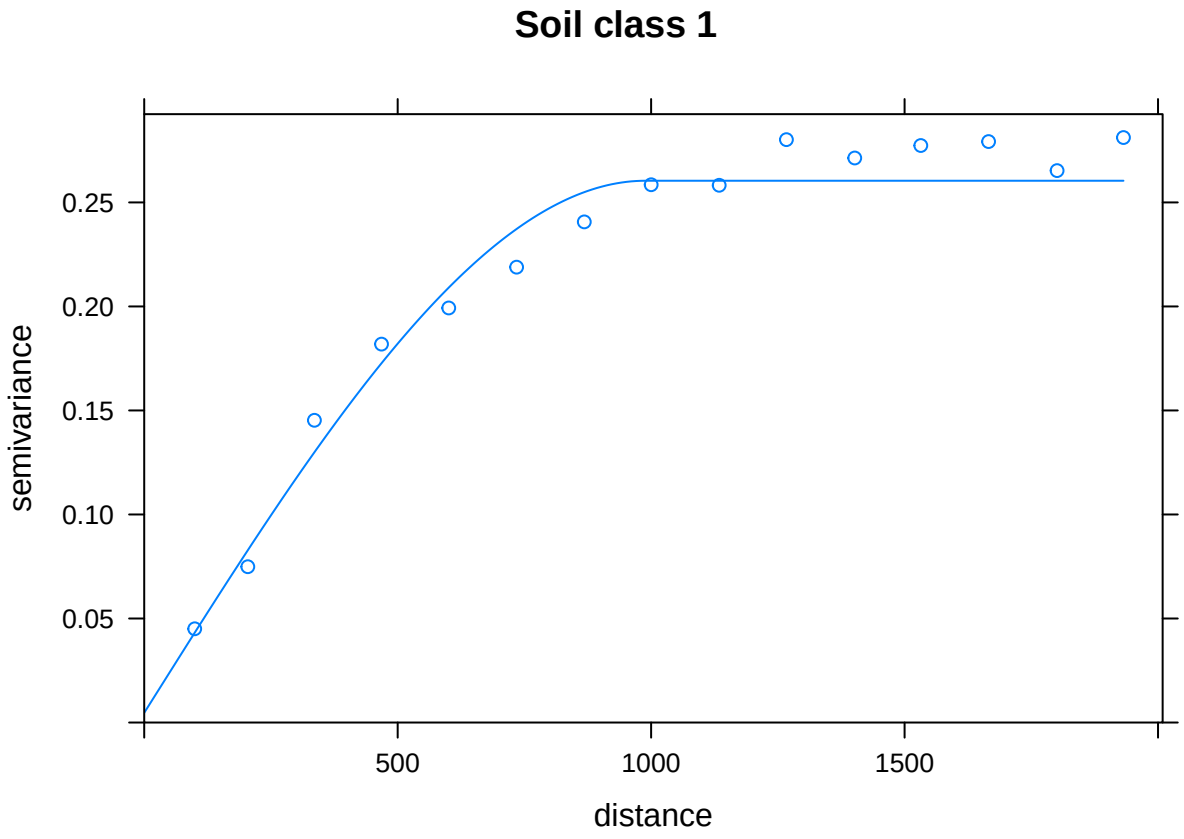
```
plot(meuse["soil"], pch = 16, reset = FALSE)
plot(meuse.riv, add = TRUE, col = NA)
```



Las categorías individuales se pueden interpolar a nuevas ubicaciones utilizando el indicador kriging. Como antes, el procedimiento comienza haciendo y modelizando el variograma, para posteriormente interpolarlo a una cuadrícula.

Comience haciendo y modelando el variograma de muestra. Tenga en cuenta que en lugar de crear una nueva variable, usamos la función `I ()`, que le dice a R que cree una nueva variable internamente en un modelo, aquí un valor binario donde la clase de suelo 1 es igual a 1, y otras clases son iguales a cero:

```
s1.var <- variogram(I(soil == 1) ~ 1, meuse, cutoff = 2000)

s1.vgm <- vgm(psill = 0.25, model = "Sph", range = 900, nugget = 0.1)

s1.vgm <- fit.variogram(s1.var, s1.vgm)

plot(s1.var, s1.vgm, main = "Soil class 1")
```

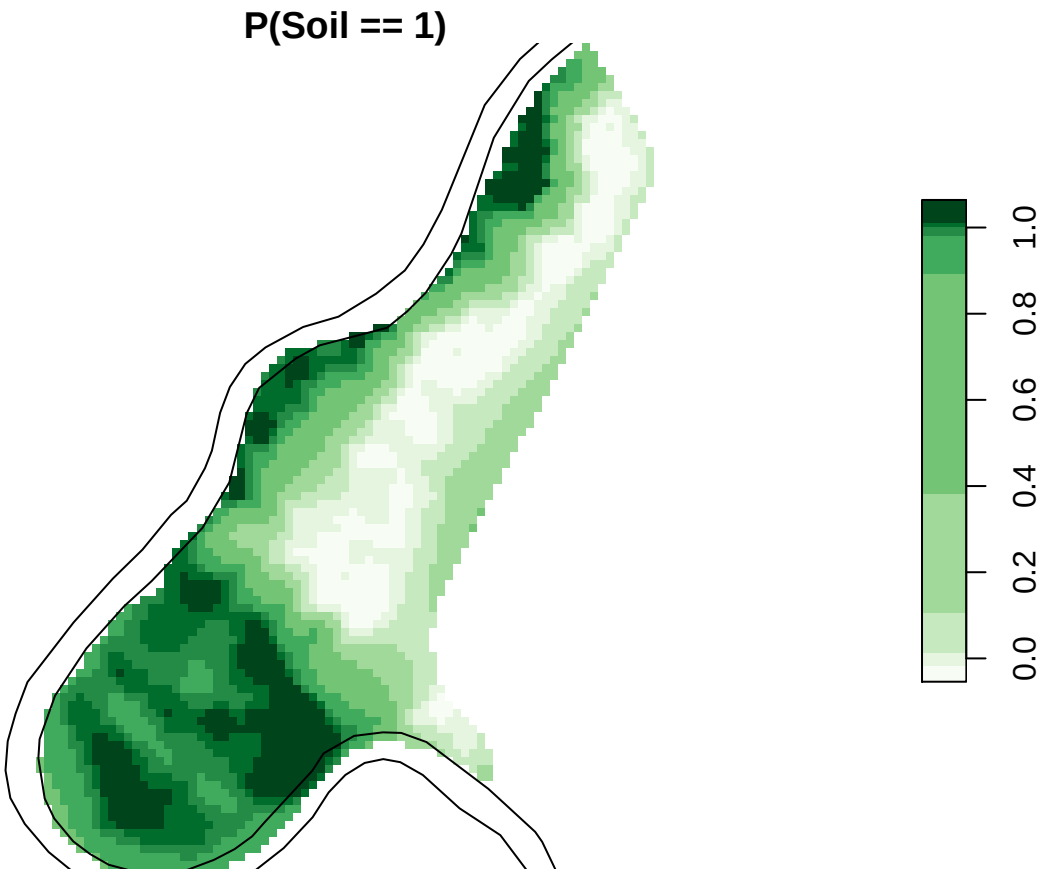


A continuación interpolamos usando la función `krige()`:

```
s1.ik <- krige(I(soil == 1) ~ 1, meuse, meuse.grid, s1.vgm)
```

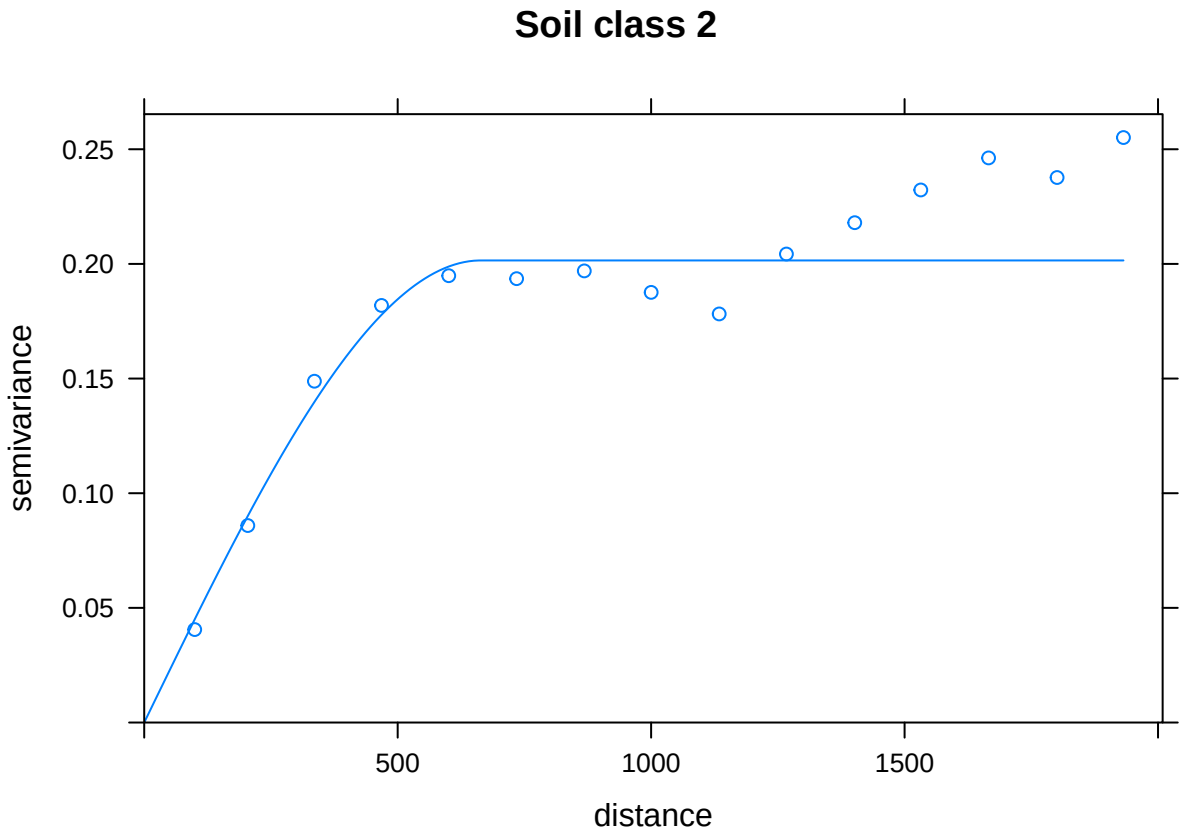
```
## [using ordinary kriging]
```

```
my.pal <- brewer.pal(9, "Greens")

plot(s1.ik["var1.pred"],
     col = my.pal,
     main = "P(Soil == 1)",
     reset = FALSE)

plot(meuse.riv, col = NA, add = TRUE)
```



Podemos hacer lo mismo con las otras dos clases de suelo, para obtener probabilidades interpoladas correspondientes a las clases 2 (`s2.ik`) y (`s3.ik`).

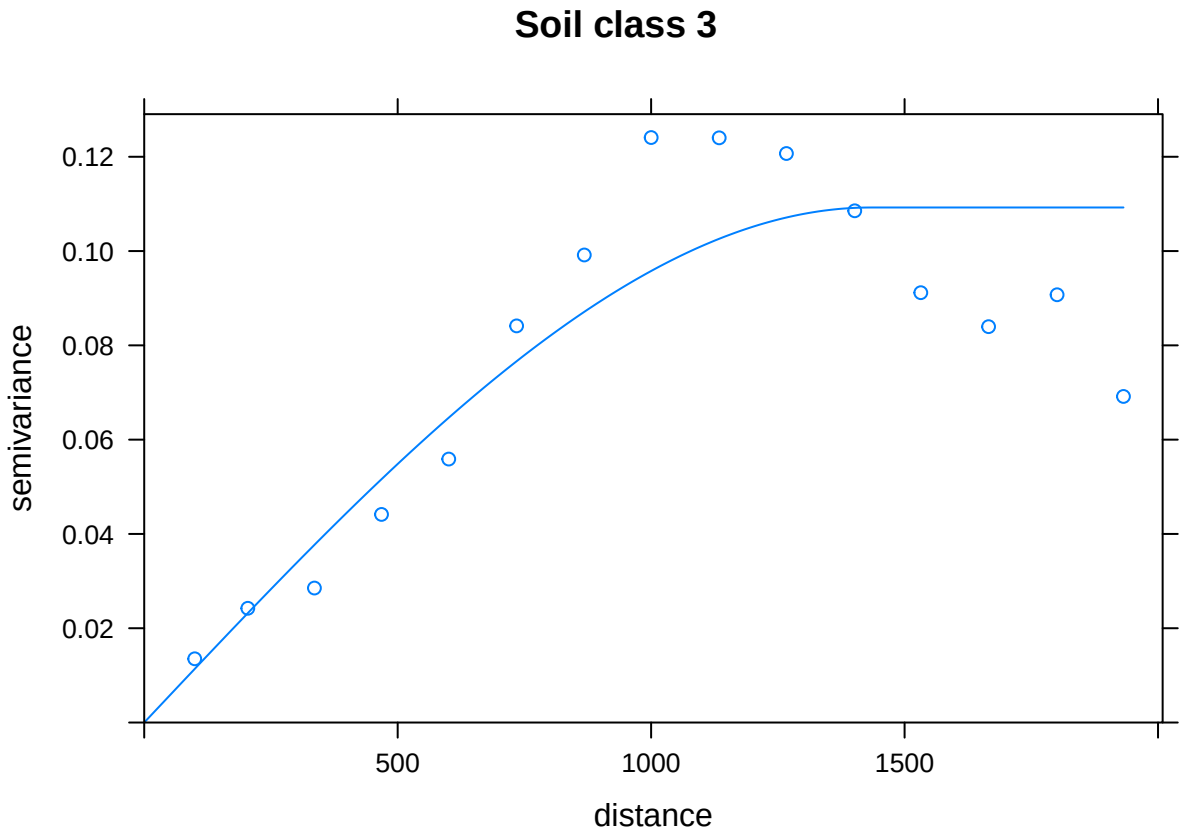
```
s2.var <- variogram(I(soil == 2) ~ 1, meuse, cutoff = 2000)

vgm_model <- vgm(psill = 0.25, model = "Sph", range = 900, nugget = 0.1)

s2.vgm <- fit.variogram(s2.var, model = vgm_model)

plot(s2.var, s2.vgm, main = "Soil class 2")
```




```
s2.ik <- krige(I(soil == 2) ~ 1, meuse, meuse.grid, s2.vgm)
```

```
## [using ordinary kriging]
```

```
s3.var <- variogram(I(soil == 3) ~ 1,
                    meuse,
                    cutoff = 2000)

s3.vgm <- fit.variogram(s3.var, model = vgm_model)

plot(s3.var, s3.vgm, main = "Soil class 3")
```

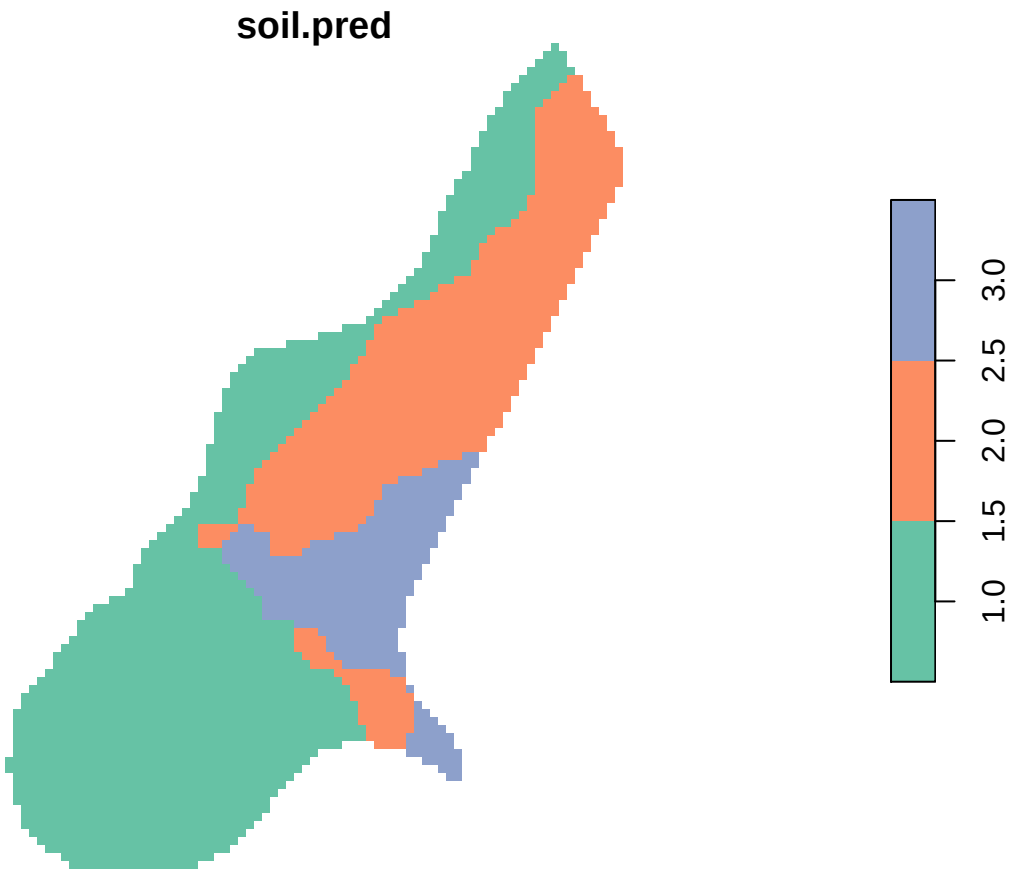



```
s3.ik <- krige(I(soil == 3) ~ 1, meuse, meuse.grid, s3.vgm)
```

```
## [using ordinary kriging]
```

Una vez que tengamos las probabilidades para cada una de las tres clases, podemos usarlas para estimar la clase más probable en cada nueva ubicación. Hacemos esto en tres pasos: + Primero, combinamos las interpolaciones de probabilidad individuales en una única matriz; + Segundo, en cada fila buscamos la columna con mayor probabilidad usando `max.col ()`, creando una nueva variable con esas salidas en `meuse.grid`; + Tercero, cartografiamos los resultados.

```
soil.prob <- cbind(c(s1.ik$var1.pred),
                   c(s2.ik$var1.pred),
                   c(s3.ik$var1.pred))

meuse.grid$soil.pred <- max.col(soil.prob)

my.pal <- brewer.pal(3, "Set2")

plot(meuse.grid["soil.pred"], col = my.pal)
```



Esto se puede extender a un mayor número de clases, siempre que haya datos suficientes para producir un variograma para cada una y se tenga la paciencia para modelarlas todas.

5. SIMULACIÓN GEOESTADÍSTICA

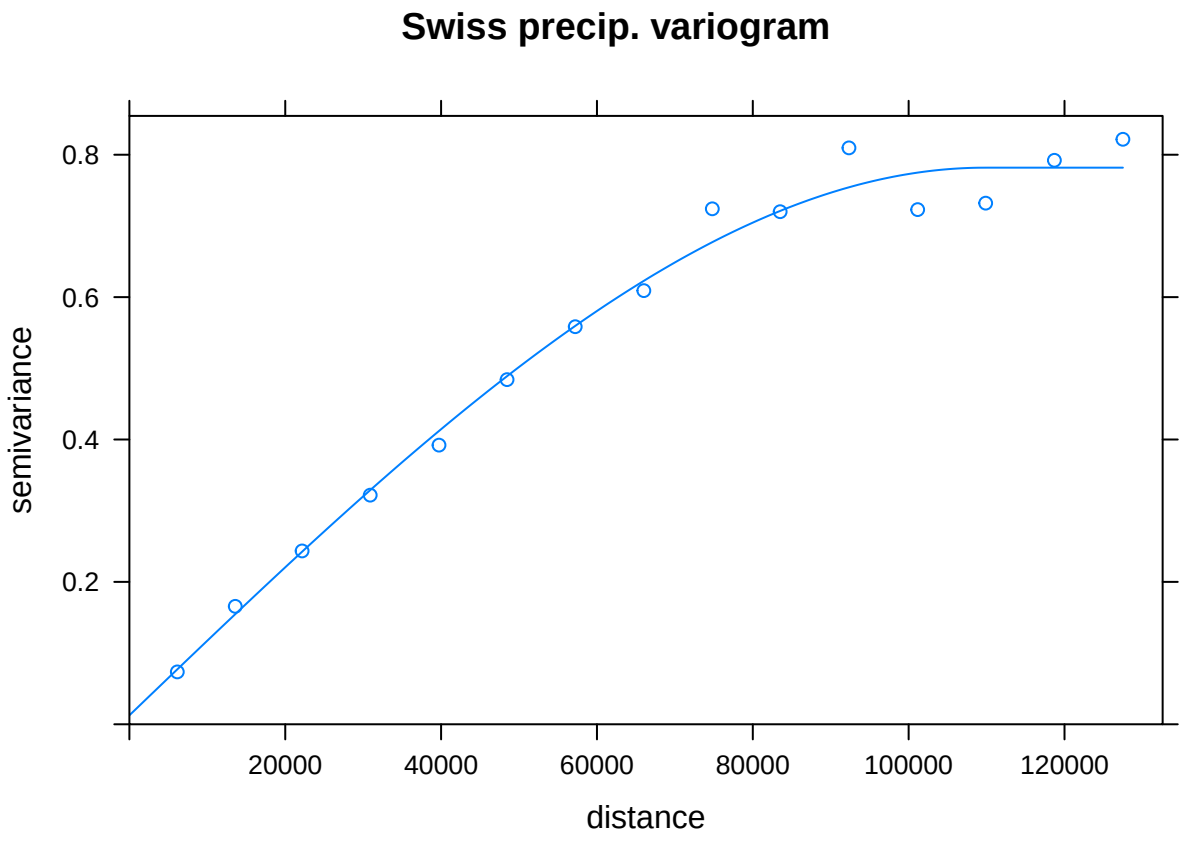
Todos los métodos de simulación geoestadística están diseñados para producir campos espaciales aleatorios, donde el valor en cada ubicación se obtiene mediante extracciones aleatorias de una función de distribución de probabilidad definida por las observaciones. En contraste con la generación directa de valores aleatorios, los campos espaciales aleatorios producen valores aleatorios en cada ubicación, pero preservando la estructura espacial. Las simulaciones individuales son mucho menos suaves que la interpolación de kriging, ya que los valores en dos ubicaciones vecinas se eligen al azar, pero están correlacionados espacialmente como se describe en un variograma.

Las simulaciones geoestadísticas se presentan en dos formas: restringidas y no restringidas. + En el tipo no restringido, el campo aleatorio se basa en una media y varianza especificadas, y el variograma para la estructura espacial. Los campos aleatorios producidos tienen las mismas características estadísticas y espaciales, pero los mínimos y máximos pueden ocurrir en cualquier parte del área de estudio. + Las simulaciones restringidas también incluyen la ubicación y el valor de los puntos observados. Esto asegura que los mínimos y máximos ocurren donde están definidos por los puntos originales, y los campos resultantes tienen el mismo patrón que los datos originales. Por esta razón la sesión se concentrará en las simulaciones restringidas.

5.1 Simulación gaussiana

Al igual que el kriging ordinario, la simulación gaussiana se puede realizar con variables continuas y utiliza una configuración similar al kriging en sesiones anteriores. El siguiente ejemplo utiliza los datos de precipitación de Suiza. El procedimiento se inicia con la obtención de un variograma y el ajuste de un modelo de variograma.

```
modNugget <- 0.05
modRange <- 100000
modSill <- 0.75

ppt.var <- variogram(lppt ~ 1, swiss.sf)

ppt.vgm1 <- vgm(psill = modSill,
                "Sph",
                range = modRange,
                nugget = modNugget)

ppt.vgm2 <- fit.variogram(ppt.var, ppt.vgm1)

plot(ppt.var, ppt.vgm2, main = "Swiss precip. variogram")
```



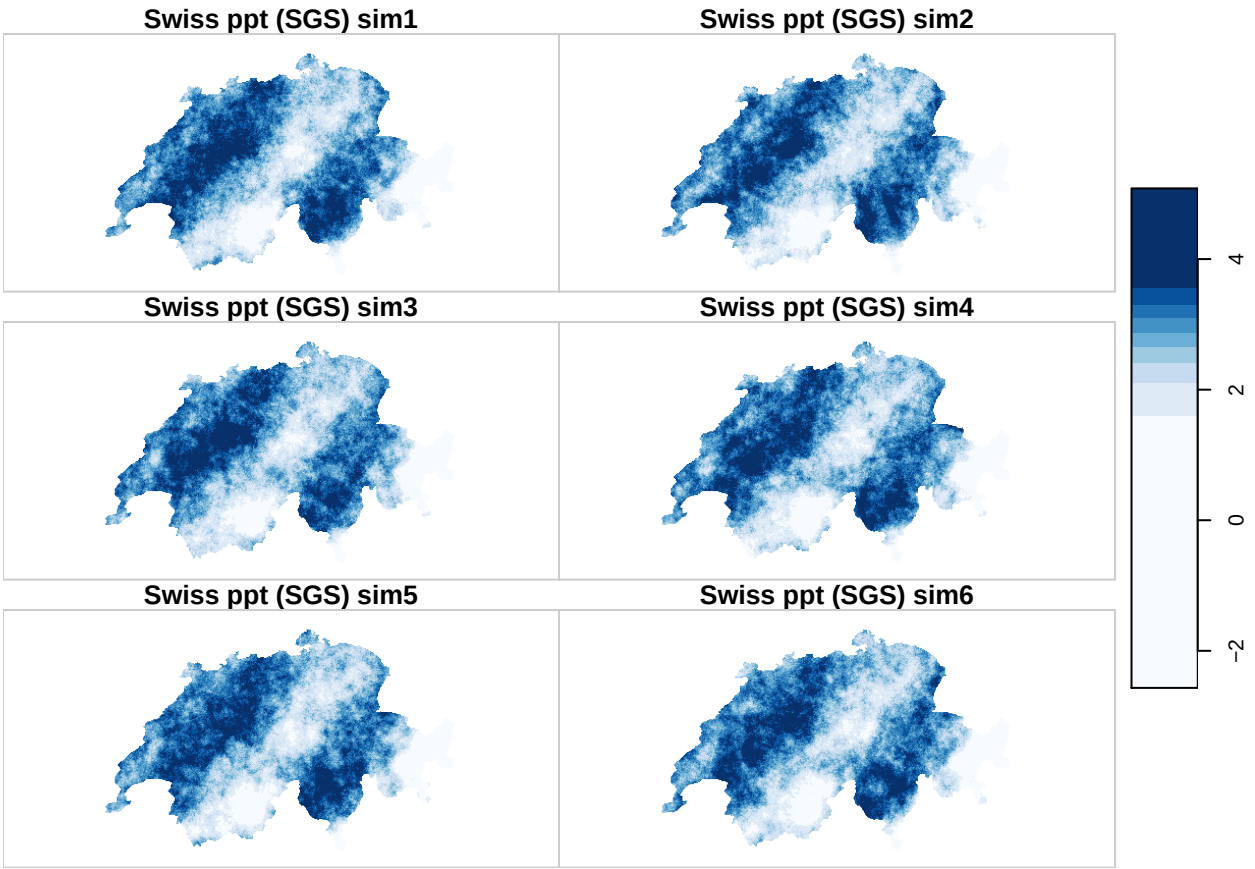
Para realizar 6 simulaciones aleatorias de los datos de precipitación, usamos la función `krige ()` nuevamente, con los datos espaciales, cuadrícula de salida, variograma, etc. El nuevo parámetro usado aquí es `nsim` que controla el número de simulaciones:

```
ppt.pred.sgs <- krige(lppt ~ 1,
                     swiss.sf,
                     swiss.dem,
                     ppt.vgm2,
                     nmax = 40,
                     nsim = 6)
```

```
## drawing 6 GLS realisations of beta...
## [using conditional Gaussian simulation]
```

El paquete `spplot ()` es muy útil para visualizar la salida, ya que efectua un mapa con los resultados de todas las simulaciones:

```
my.pal <- brewer.pal(9, "Blues")

plot(ppt.pred.sgs,
     col = my.pal,
     main = "Swiss ppt (SGS)")
```



En general, las simulaciones individuales solo son de interés para examinar el grado de variación entre observaciones (a diferencia del kriging, que proporciona interpolaciones suavizadas). El poder de este enfoque de simulación reside en su capacidad para producir una gran cantidad de posibles realizaciones de un mismo campo espacial. Esto permite una mejor evaluación de la incertidumbre en cualquier ubicación, ya que podemos obtener no solo el valor medio estimado y un intervalo de confianza, sino la distribución de probabilidad completa de los valores interpolados.

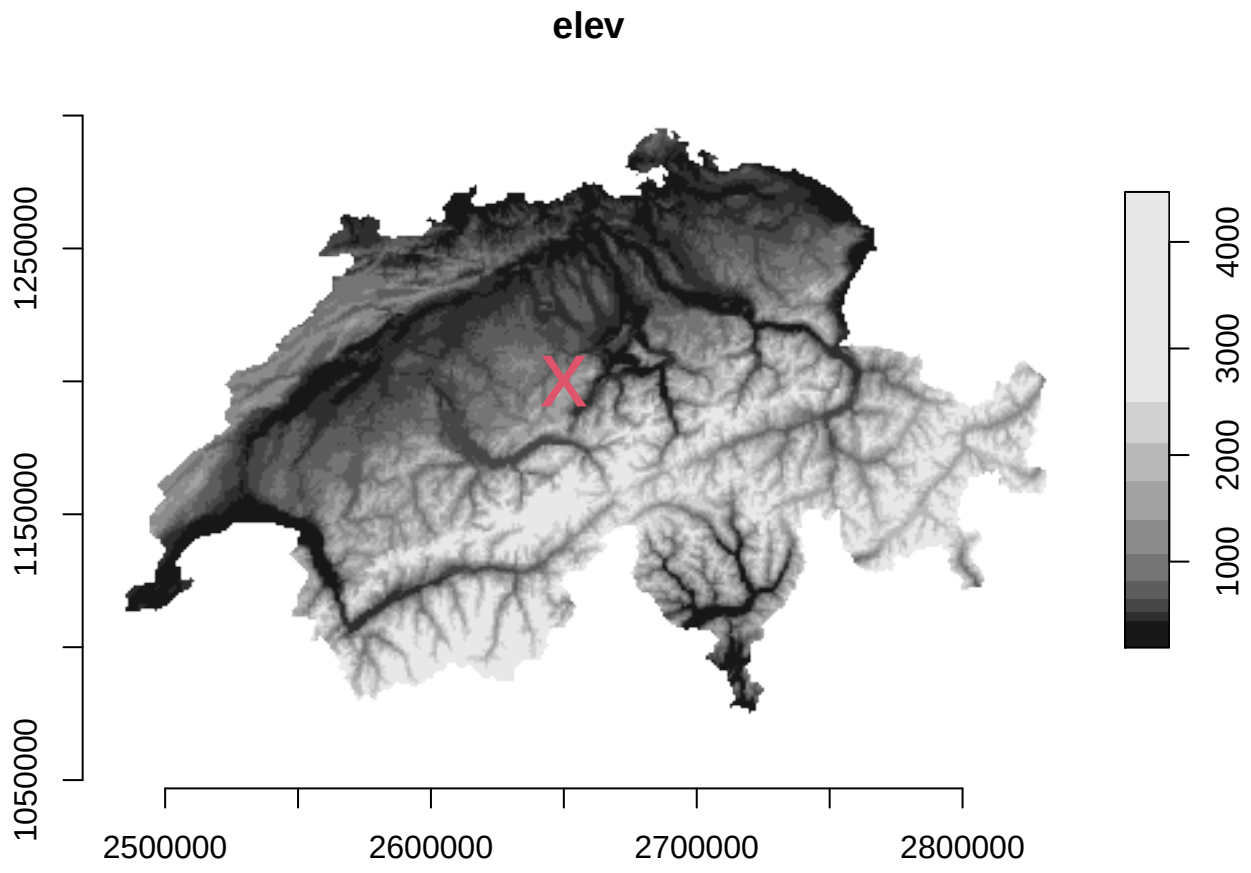
En el siguiente fragmento de código, produciremos cien simulaciones de precipitación, luego extraeremos los valores predichos para un solo punto y lo convertiremos en un histograma. Para ello, se vuelve a ejecutar la función `krige` con un mayor número de simulaciones:

```
ppt.pred.sgs <- krige(lppt ~ 1,
                      swiss.sf,
                      swiss.dem,
                      ppt.vgm2,
                      nmax = 40,
                      nsim = 100)
```

```
## drawing 100 GLS realisations of beta...
## [using conditional Gaussian simulation]
```

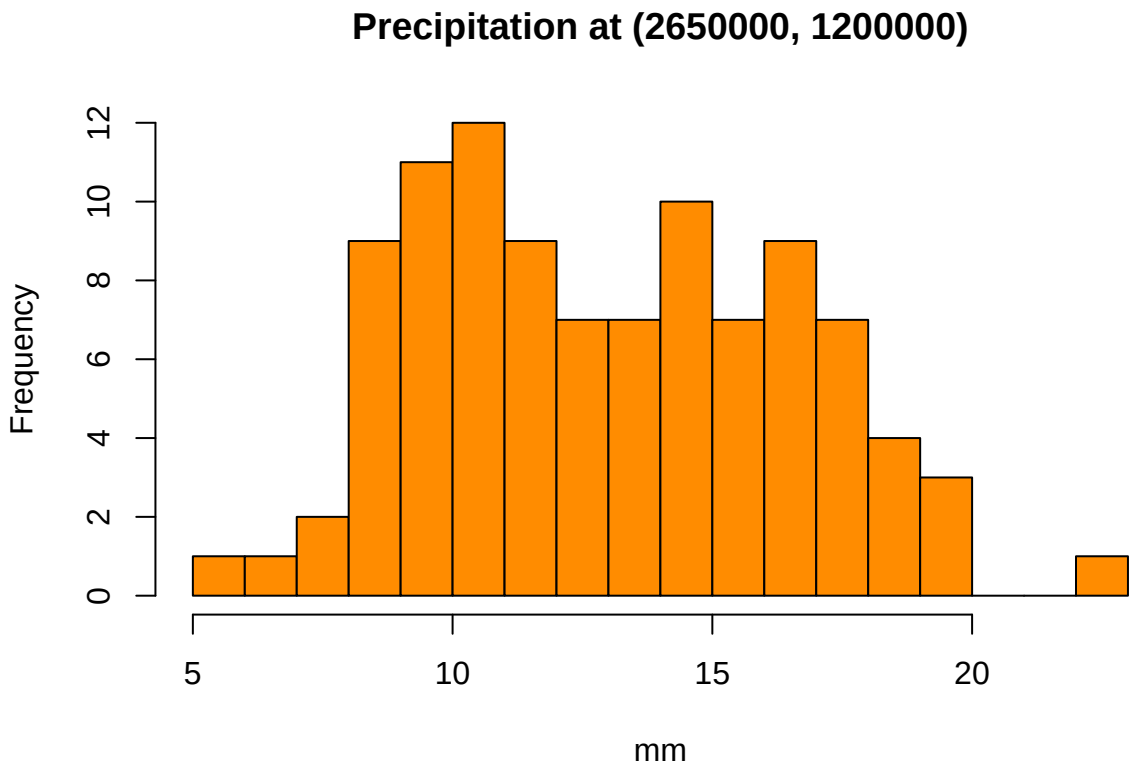
A continuación, se crea una localización artificial para extraer los valores simlados.

```
newloc <- st_geometry(st_point(c(2650000, 1200000)))

st_crs(newloc) <- st_crs(swiss.dem)

plot(swiss.dem, reset = FALSE, axes = TRUE)

plot(newloc, pch = "x", cex = 3, col = 2, add = TRUE)
```



Con la función `st_extract()` del paquete `sf` se extraen los valores correspondientes a ese punto, que van a ser representados gráficamente mediante un histograma:

```r
newloc.ppt <- st_extract(ppt.pred.sgs, newloc)

newloc.ppt$ppt <- exp(newloc.ppt$var1)

hist(newloc.ppt$ppt,
     breaks = 20,
     col = "darkorange",
     main = "Precipitation at (2650000, 1200000)",
     xlab = "mm")
```



Debe tenerse en cuenta que, como estamos usando la misma notación de fórmula que kriging, podríamos extender fácilmente el enfoque de simulación básico para incluir covariables para representar desviaciones o tendencias externas.

5.2 Simulación de indicador

El enfoque de simulación también se puede utilizar para la interpolación de indicadores. Como en el ejemplo anterior, simplemente reutilizamos la función `krige ()`. Además del parámetro "nsim", incluimos como argumento `indicators=TRUE` para aplicar kriging al indicador.

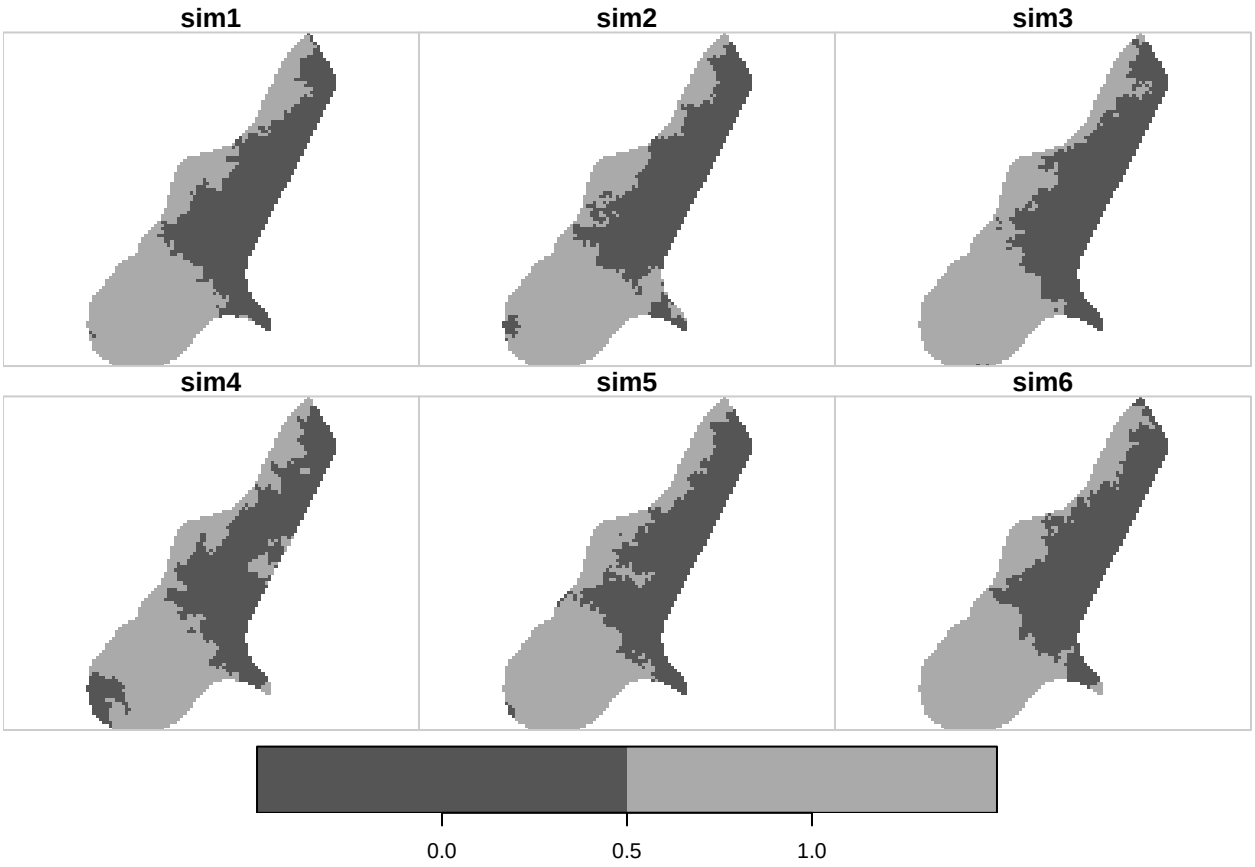
En este caso, en lugar de que cada simulación estime una probabilidad en cada nueva ubicación, el procedimiento estima un valor binario (presencia o ausencia) basándose en una distribución binomial.

Como ejemplo, para simular 6 realizaciones de la distribución de la clase de suelo 1:

```
s1.sis <- krige(I(soil == 1) ~ 1,
                meuse,
                meuse.grid,
                s1.vgm,
                nsim = 6,
                indicators = TRUE,
                nmax = 40)
```

```
## drawing 6 GLS realisations of beta...
## [using conditional indicator simulation]
```

```
plot(s1.sis)
```

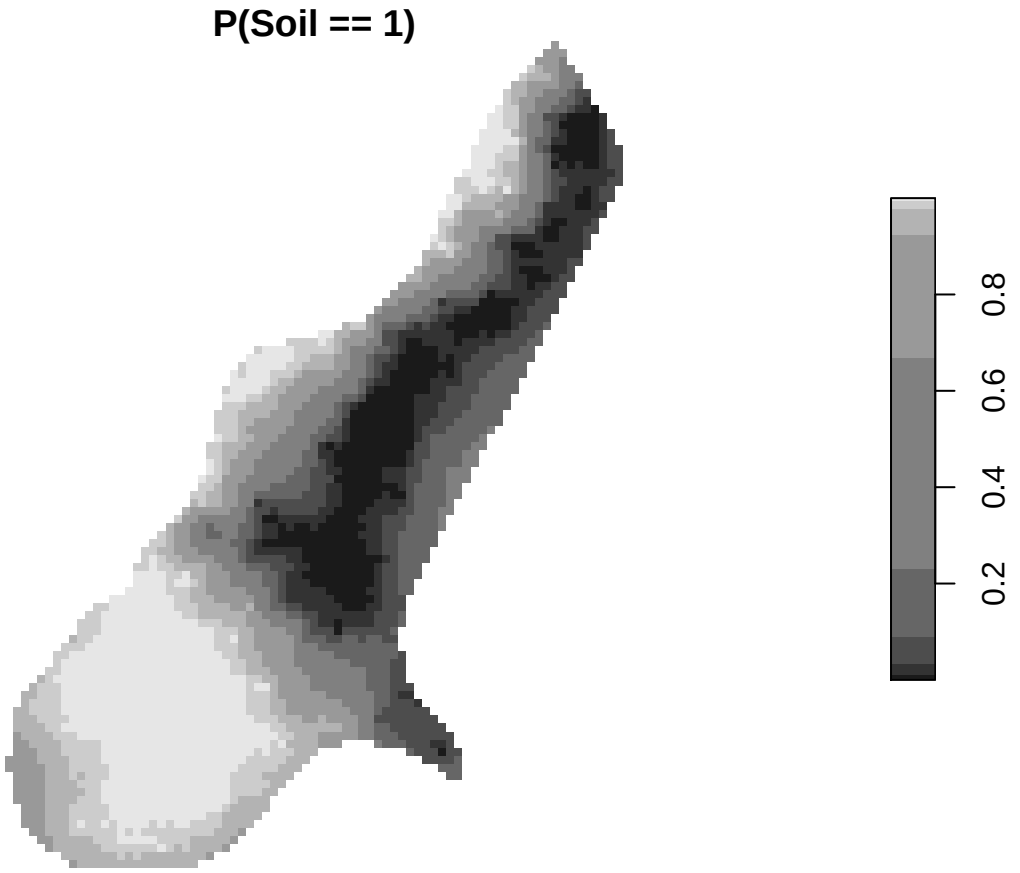


Si ahora ejecutamos múltiples simulaciones, podemos evaluar la incertidumbre. En este caso, obtenemos 1000 simulaciones de la presencia del tipo de suelo 1 en cada ubicación. Para estimar su probabilidad, tomamos la suma de todas las presencias (1) y la dividimos por el número de simulaciones (1000). Esto se almacena en la salida y luego se puede trazar usando `spplot ()`.

```
s1.sim <- krige(I(soil == 1) ~ 1,
                meuse,
                meuse.grid,
                s1.vgm,
                nsim = 1000,
                indicators = TRUE,
                nmax = 40)
```

```
## drawing 1000 GLS realisations of beta...
## [using conditional indicator simulation]
```

```
s1.prob <- st_apply(s1.sim, c(1,2), sum) / 1000

plot(s1.prob, main = "P(Soil == 1)")
```



En contraste con el indicador kriging, la salida de esta función proporciona verdaderas probabilidades de presencia, restringidas al rango [0,1].

6. EJERCICIOS

El fichero shapefile *ozone.shp*, que se encuentra en el archivo comprimido *mwozone.zip* contiene mediciones de la concentración de ozono en el medio oeste de los EE.UU correspondientes al 20 de junio de 1987.
El objetivo de este ejercicio es, a partir de esos datos, cartografiar la probabilidad de que se supere un cierto umbral (100 ppb) en esa fecha. Puede hacer esto usando kriging de indicador o simulación de indicador, usando los ejemplos proporcionados en el laboratorio. De forma más extensa, debe incluirse ademas:

- Un mapa de concentraciones de ozono en las estaciones de medición.
- Un gráfico del variograma de muestra que describe la estructura espacial de la condición que desea interpolar (ozono $>$ 100 ppb) y una breve descripción de la estructura espacial que observa.
- Un modelo de variograma ajustado (como una figura), proporcionando el modelo elegido, el umbral, el rango y el valor de la pepita (si corresponde)

- Un mapa de probabilidades pronosticadas para superar el nivel de 100 ppb de ozono. Para hacer esto, necesitará crear una cuadrícula de longitud/latitud para la interpolación. Consulte el código a continuación para saber cómo hacer esto simplemente en R
- Una breve descripción del patrón espacial que se muestra en el mapa interpolado ¿Con qué crees que se relacionan las probabilidades más altas?

Código para añadir los shapefiles

Puede ser útil agregar otros shapefiles a sus mapas para este ejercicio. El siguiente código asume que ha leído en el archivo de forma de ozono un objeto `sf` llamado`ozone.sf`, y agrega contornos estatales, lagos y ciudades usando los Natural Earth datasets. El ejemplo propuesto traza las concentraciones de ozono, pero puede adaptarse con bastante facilidad para los valores predichos que obtenga. Tenga en cuenta que para ejecutar este código, deberá asegurarse de que el paquete **tmap** esté instalado.

- *ne_50m_admin_1_states_provinces.zip*
- *ne_50m_lakes.zip*
- *ne_50m_populated_places.zip*

Para superponer múltiples shapefiles, podemos hacer uso de los argumentos `reset` y`add` al trazar objetos `sf`. Comenzaremos leyendo la capa de ozono, y luego las 3 tres capas adicionales:

```
ozone.sf <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/mwozone/ozone.shp")
st_crs(ozone.sf) <- 4326
```

```
## Importación de los datos
states <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/ne_50m_admin_1_states_provinces/ne_50m_admi
lakes <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/ne_50m_lakes/ne_50m_lakes.shp")
places <- st_read("D:/Docencia_Master_2021/GEOG6000/sesion05/ne_50m_populated_places/ne_50m_populated_pl
```

A continuación se trazarán las capas. Cada vez que agregamos el argumento `reset = TRUE` le informamos a R que mantenga el mismo esquema básico de la trama. Para el segundo (y los gráficos posteriores) también establecemos `add = TRUE` para evitar que R reinicie el gráfico.

```
plot(st_geometry(ozone.sf), reset = FALSE)
plot(st_geometry(lakes), reset = FALSE, add = TRUE, col = "lightblue")
plot(st_geometry(states), reset = FALSE, add = TRUE)
plot(ozone.sf["ozone"], add = TRUE, pch = 16)
```

Se pueden hacer mejores gráficos con capas múltiples usando la función `geom_sf` de **ggplot2**:

```
places <- cbind(places, st_coordinates(st_centroid(places)))

ggplot() +
  geom_sf(data = lakes, fill = "lightblue") +
  geom_sf(data = states, fill = NA) +
  geom_sf(data = ozone.sf, aes(col = ozone), size = 2) +
  scale_color_viridis_c() +
  geom_label(data = places, aes(X, Y, label = NAME), size = 2.5) +
  coord_sf(xlim = c(-94, -82), ylim = c(36, 45), expand = FALSE) +
  theme_bw()
```

Alternativamente, se puede usar el paquete **tmap** para una imagen de mejor calidad

```r
library(tmap)

mybbox <- st_bbox(ozone.sf)

tm_shape(lakes, bbox = mybbox) +
  tm_fill("lightblue") +
  tm_shape(states) +
  tm_borders() +
  tm_shape(ozone.sf) +
  tm_symbols(col = "ozone", size = 0.5, palette = "viridis") +
  tm_shape(places) +
  tm_text("NAME", size = 0.75)
```

Código para diseñar una rejilla para predicciones

```r
pred.grid <- st_as_stars(mybbox,
                         xlim = c(-94, -82),
                         ylim = c(36, 45),
                         dx = 0.1,
                         dy = 0.1)

st_crs(pred.grid) <- 4326

plot(pred.grid)
```