Modelos para datos temporales y espaciales.

Trabajo Tema 5. Modelos espaciales.

Jerónimo Carranza Carranza

29 de agosto de 2017

Contents

1 Formulación

Opción 2. Aplicar técnicas de krigeado a una de las variables logaritmo de la concentración de cadmio, cobre o plomo del conjunto de datos "meuse", realizando la predicción sobre el conjunto pixelado "meuse.grid". En concreto, realizar los pasos:

- Descripción de la variable (resumen y representaciones gráficas)
- CONSTRUCCIÓN DEL VARIOGRAMA MUESTRALY AJUSTE A UN MODELO TEÓRICO de la variable objetivo
- KRIGING ORDINARIO PARA LA VARIABLE OBJETIVO
- KRIGING UNIVERSAL PARA LA VARIABLE OBJETIVO y comparar con los resultados obtenidos en el paso anterior.
- KRIGING DERIVA EXTERNA PARA LA VARIABLE OBJETIVO CON PREDICTOR DISTANCIA AL RIO y comparar con los resultados obtenidos pasos anteriores.
- KRIGING RESIDUAL DIRECTO PARA LA VARIABLE OBJETIVO CON PREDICTOR DISTANCIA AL RIO y comparar con los resultados obtenidos pasos anteriores.

2 Carga de librerías

```
library(sp)
library(lattice)
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(gstat)
```

3 Lectura de datos

```
data(meuse)
class(meuse)
```

```
## [1] "data.frame"
```

```
dim(meuse)
```

```
## [1] 155  14
```

```
names(meuse)
```

```
##  [1] "x"       "y"       "cadmium" "copper"  "lead"    "zinc"    "elev"
##  [8] "dist"    "om"      "ffreq"   "soil"    "lime"    "landuse" "dist.m"
```

```
head(meuse)
```

```
##        x      y cadmium copper lead zinc  elev       dist   om ffreq soil
## 1 181072 333611    11.7     85  299 1022 7.909 0.00135803 13.6     1    1
```

```
## 2 181025 333558     8.6     81  277 1141 6.983 0.01222430 14.0     1    1
## 3 181165 333537     6.5     68  199  640 7.800 0.10302900 13.0     1    1
## 4 181298 333484     2.6     81  116  257 7.655 0.19009400  8.0     1    2
## 5 181307 333330     2.8     48  117  269 7.480 0.27709000  8.7     1    2
## 6 181390 333260     3.0     61  137  281 7.791 0.36406700  7.8     1    2
##   lime landuse dist.m
## 1    1      Ah     50
## 2    1      Ah     30
## 3    1      Ah    150
## 4    0      Ga    270
## 5    0      Ah    380
## 6    0      Ga    470
```

4 Conversión a objeto espacial

```
coordinates(meuse) = ~x+y
class(meuse)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
names(meuse)
```

```
##  [1] "cadmium" "copper"  "lead"    "zinc"    "elev"    "dist"    "om"
##  [8] "ffreq"   "soil"    "lime"    "landuse" "dist.m"
```

5 Resumen

```
summary(meuse)
```

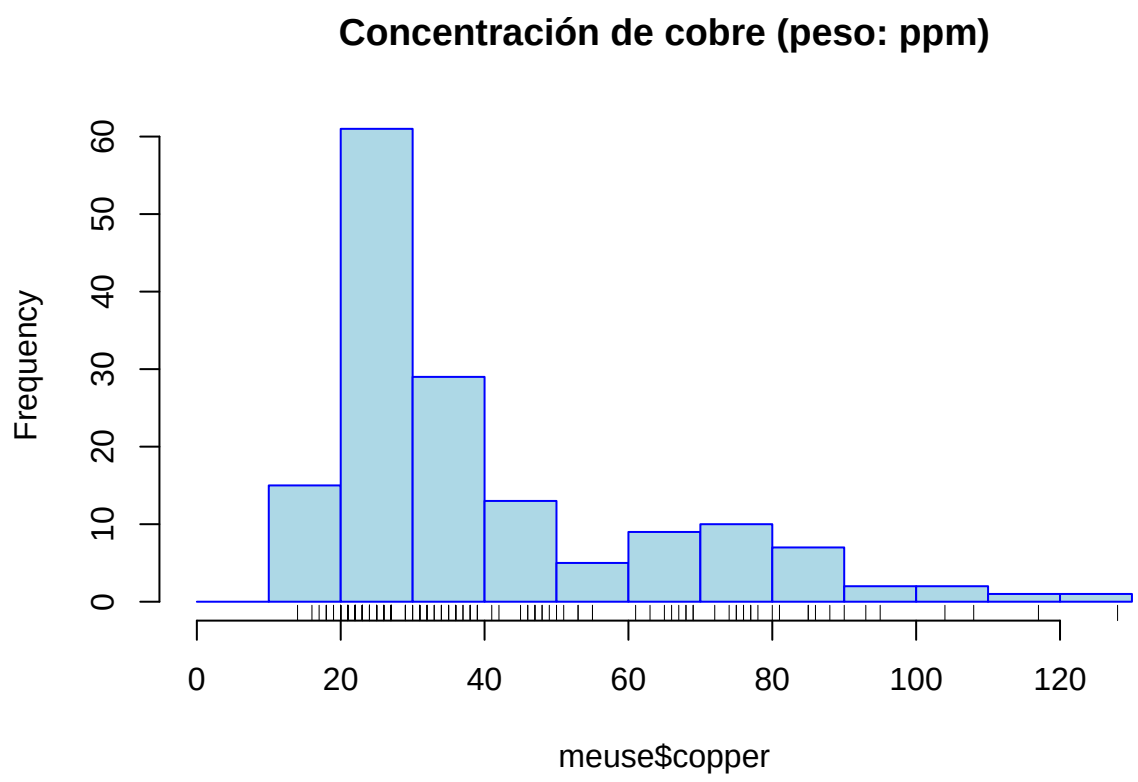
```
## Object of class SpatialPointsDataFrame
## Coordinates:
##      min    max
## x 178605 181390
## y 329714 333611
## Is projected: NA
## proj4string : [NA]
## Number of points: 155
## Data attributes:
##     cadmium           copper            lead            zinc
##  Min.   : 0.200   Min.   : 14.00   Min.   : 37.0   Min.   : 113.0
##  1st Qu.: 0.800   1st Qu.: 23.00   1st Qu.: 72.5   1st Qu.: 198.0
##  Median : 2.100   Median : 31.00   Median :123.0   Median : 326.0
##  Mean   : 3.246   Mean   : 40.32   Mean   :153.4   Mean   : 469.7
##  3rd Qu.: 3.850   3rd Qu.: 49.50   3rd Qu.:207.0   3rd Qu.: 674.5
##  Max.   :18.100   Max.   :128.00   Max.   :654.0   Max.   :1839.0
##
##       elev             dist               om         ffreq  soil   lime
##  Min.   : 5.180   Min.   :0.00000   Min.   : 1.000   1:84   1:97   0:111
##  1st Qu.: 7.546   1st Qu.:0.07569   1st Qu.: 5.300   2:48   2:46   1: 44
```

```
##  Median : 8.180   Median :0.21184   Median : 6.900   3:23   3:12
##  Mean   : 8.165   Mean   :0.24002   Mean   : 7.478
##  3rd Qu.: 8.955   3rd Qu.:0.36407   3rd Qu.: 9.000
##  Max.   :10.520   Max.   :0.88039   Max.   :17.000
##                                     NA's   :2
##     landuse       dist.m
##  W      :50   Min.   :  10.0
##  Ah     :39   1st Qu.:  80.0
##  Am     :22   Median : 270.0
##  Fw     :10   Mean   : 290.3
##  Ab     : 8   3rd Qu.: 450.0
##  (Other):25   Max.   :1000.0
##  NA's   : 1
```

Se centra el estudio en la variable concentración de Cobre (copper), con transformación logarítmica de la variable original.

6 Representaciones gráficas

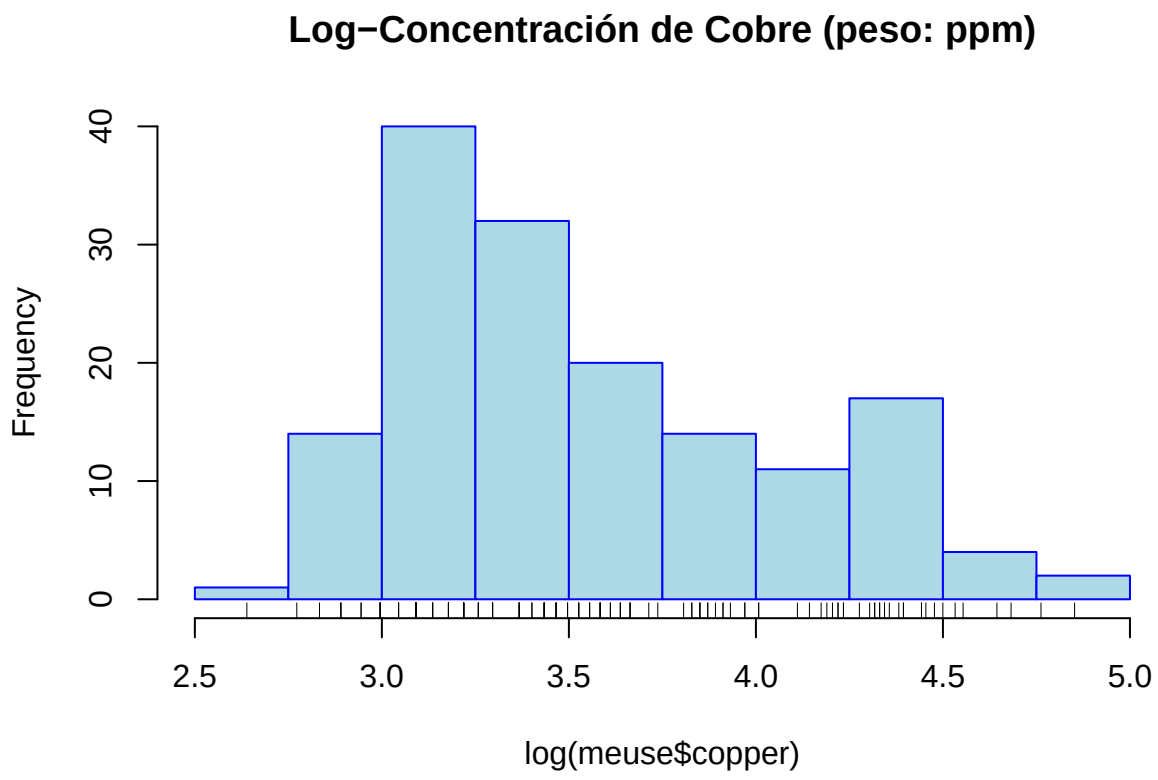
```
hist(meuse$copper, breaks = seq(0, 130, by = 10), col = "lightblue",
     border = "blue", main = "Concentración de cobre (peso: ppm)")
rug(meuse$copper)
```



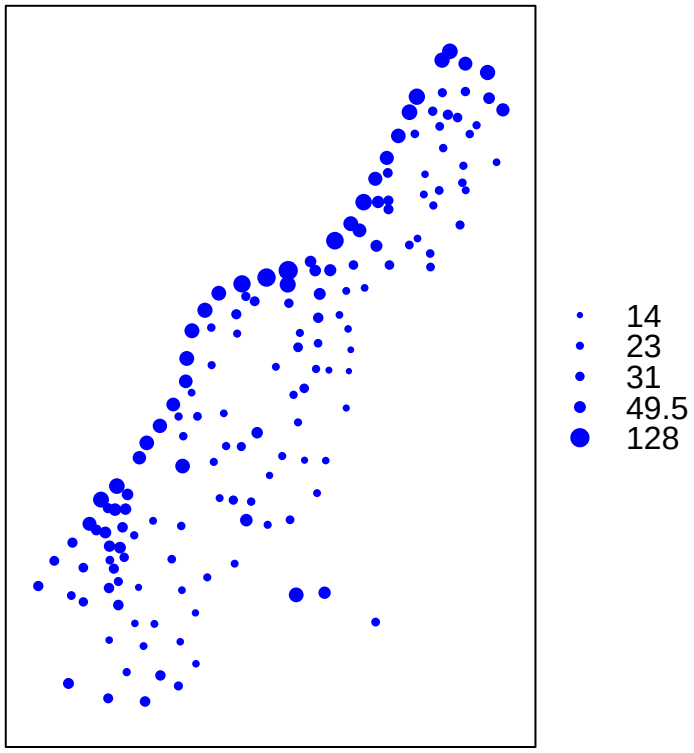

```
summary(log(meuse$copper))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.639   3.135   3.434   3.557   3.902   4.852
```

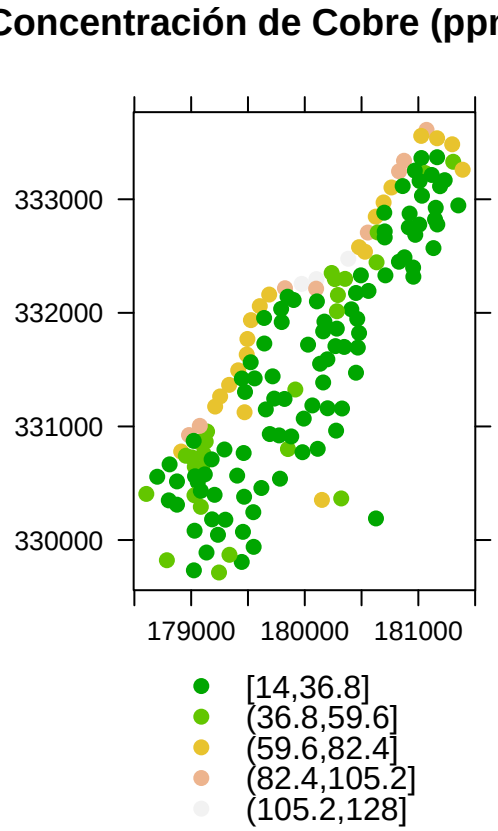
```r
hist(log(meuse$copper), breaks = seq(2.5, 5, by = 0.25), col = "lightblue",
     border = "blue", main = "Log-Concentración de Cobre (peso: ppm)")
rug(log(meuse$copper))
```




```r
bubble(meuse, c("copper"),maxsize = 1.2, col = c("red","blue"), do.sqrt = TRUE,
       main = "Concentración de Cobre (ppm)")
```




```
spplot(meuse["copper"],main="Concentración de Cobre (ppm)",
       scales=list(draw=TRUE),col.regions=terrain.colors(10))
```



7 Variables en rejilla

```
data(meuse.grid)
coordinates(meuse.grid) = ~x+y
gridded(meuse.grid) = TRUE
class(meuse.grid)
```

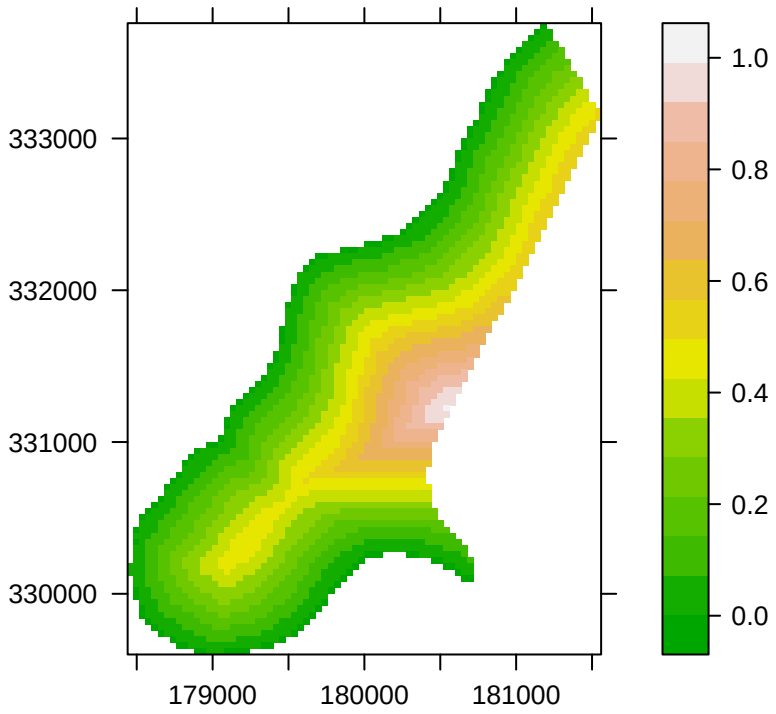
```
## [1] "SpatialPixelsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
names(meuse.grid)
```

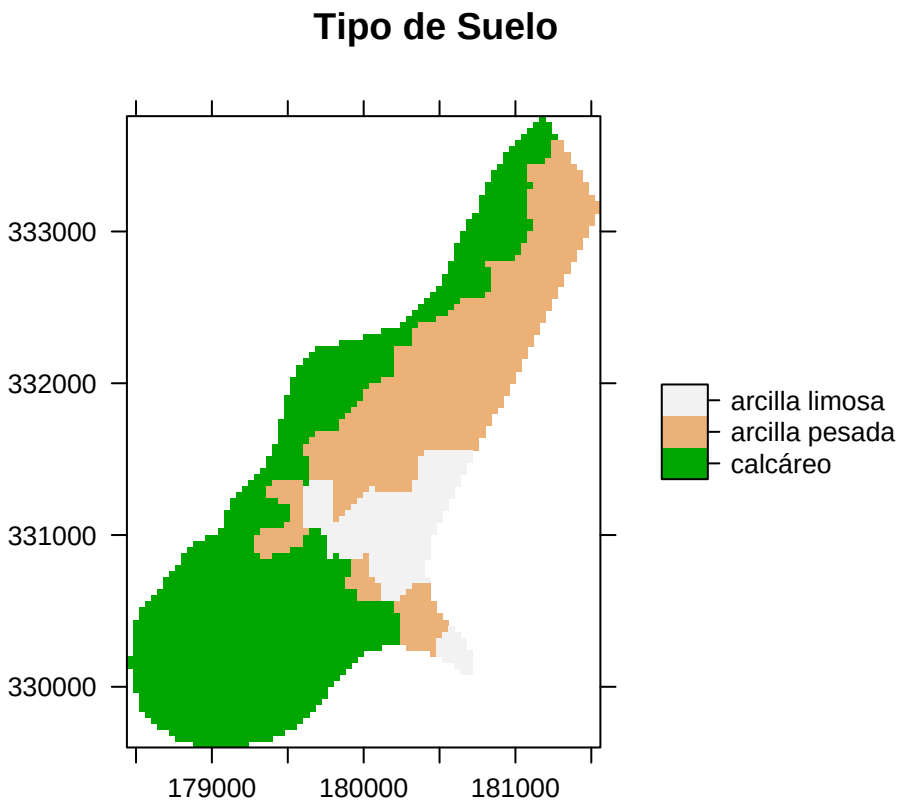
```
## [1] "part.a" "part.b" "dist"   "soil"   "ffreq"
```

```
meuse.grid$soil=factor(meuse.grid$soil,
                       labels = c('calcáreo','arcilla pesada','arcilla limosa'))
meuse.grid$ffreq=factor(meuse.grid$ffreq,
                       labels = c('cada 2 años',
                                  'cada 10 años',
                                  'cada 50 años'))
```

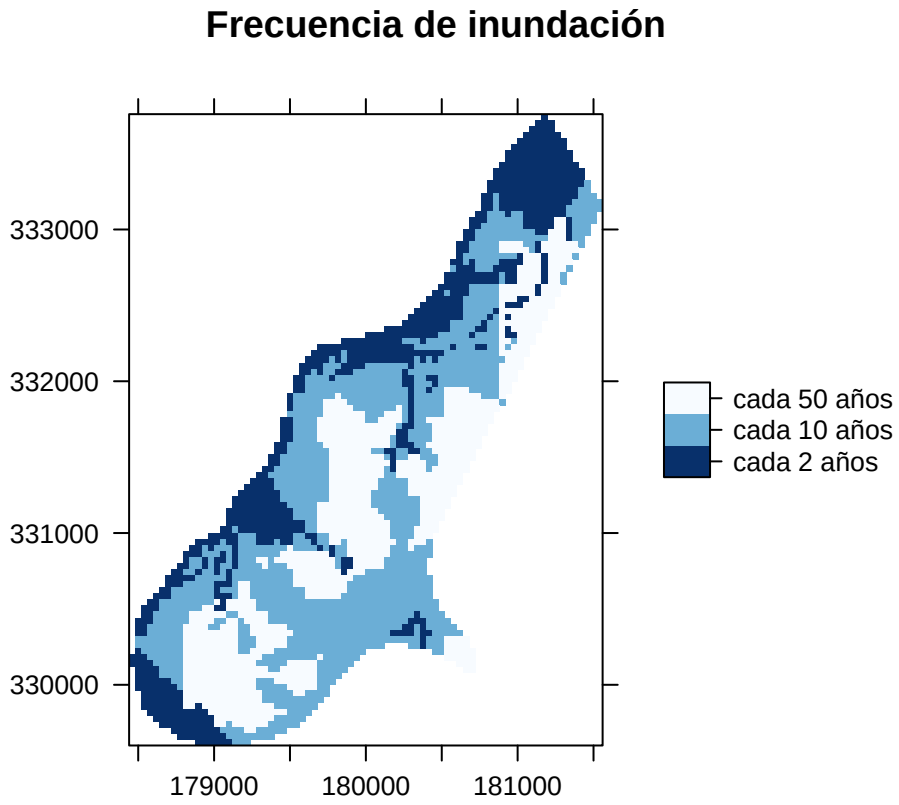
```
spplot(meuse.grid, c("dist"), col.regions=terrain.colors(20),
       main="Distancia al río", scales=list(draw=TRUE), cex=0.5)
```




```
spplot(meuse.grid, c("soil"), col.regions=terrain.colors(3),
       main="Tipo de Suelo", scales=list(draw=TRUE), cex=0.5)
```




```
spplot(meuse.grid, c("ffreq"), col.regions=rev(blues9),
       main="Frecuencia de inundación", scales=list(draw=TRUE), cex=0.5)
```

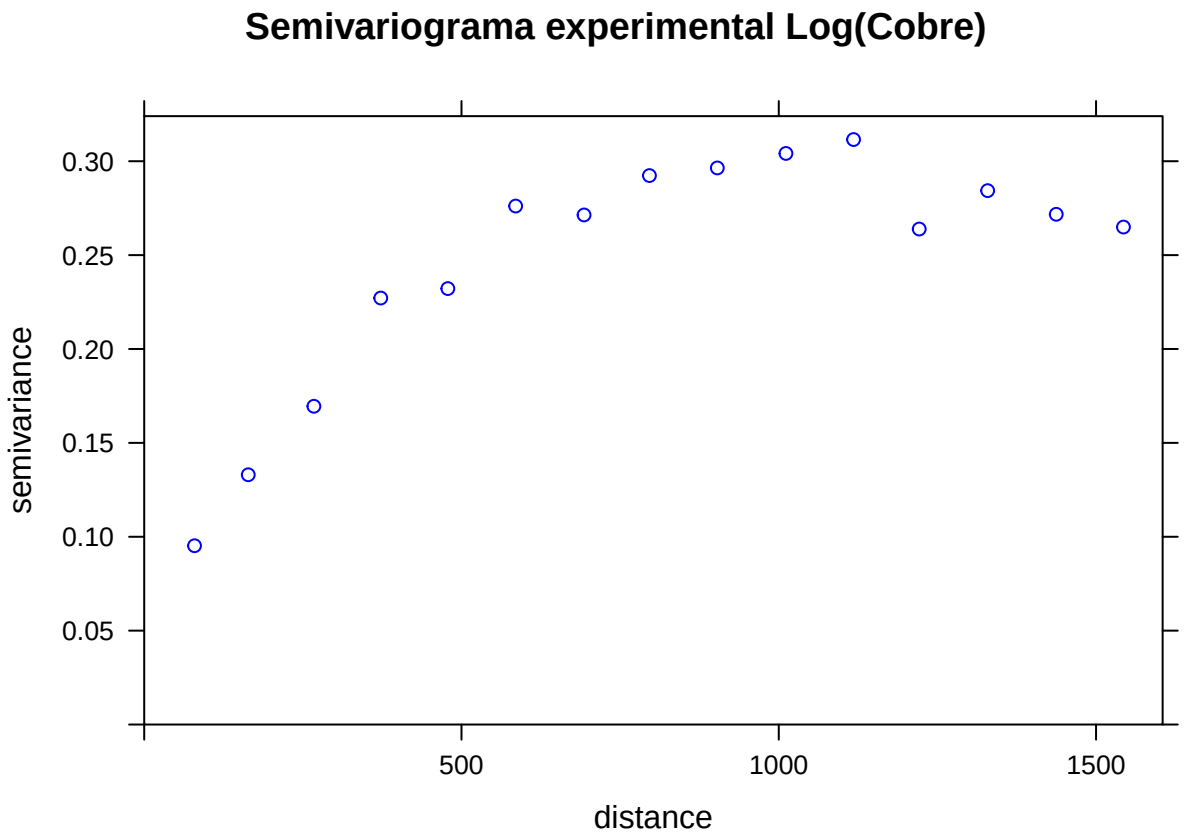


8 Variograma muestral

```
(lCu.vgm = variogram(log(copper)~1, meuse))
```

```
##     np       dist      gamma dir.hor dir.ver   id
## 1   57   79.29244 0.09522828       0       0 var1
## 2  299  163.97367 0.13301890       0       0 var1
## 3  419  267.36483 0.16949808       0       0 var1
## 4  457  372.73542 0.22712857       0       0 var1
## 5  547  478.47670 0.23217426       0       0 var1
## 6  533  585.34058 0.27613309       0       0 var1
## 7  574  693.14526 0.27139217       0       0 var1
## 8  564  796.18365 0.29236481       0       0 var1
## 9  589  903.14650 0.29642987       0       0 var1
## 10 543 1011.29177 0.30415935       0       0 var1
## 11 500 1117.86235 0.31154663       0       0 var1
## 12 477 1221.32810 0.26389361       0       0 var1
## 13 452 1329.16407 0.28434482       0       0 var1
## 14 457 1437.25620 0.27174256       0       0 var1
## 15 415 1543.20248 0.26493408       0       0 var1
```

```
plot(lCu.vgm, col="blue",main="Semivariograma experimental Log(Cobre)")
```



9 Ajuste de modelos teóricos

```
modelos = vgm()
modelos = data.frame(modelos,'SSErr'=NA)
for (i in c(2:15,17,18)) { # Se han excluido manualmente los que han dado error
    modelos$SSErr[i] = attributes(fit.variogram(
      lCu.vgm, model=vgm(0.25, modelos$short[i], 900, 0.05)))$SSErr
}
```

```
## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.vgm, model = vgm(0.25, modelos$short[i],
## 900, : No convergence after 200 iterations: try different initial values?
```

```
## Warning in fit.variogram(lCu.vgm, model = vgm(0.25, modelos$short[i],
## 900, : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.vgm, model = vgm(0.25, modelos$short[i],
## 900, : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.vgm, model = vgm(0.25, modelos$short[i],
## 900, : singular model in variogram fit
```

```
modelos
```

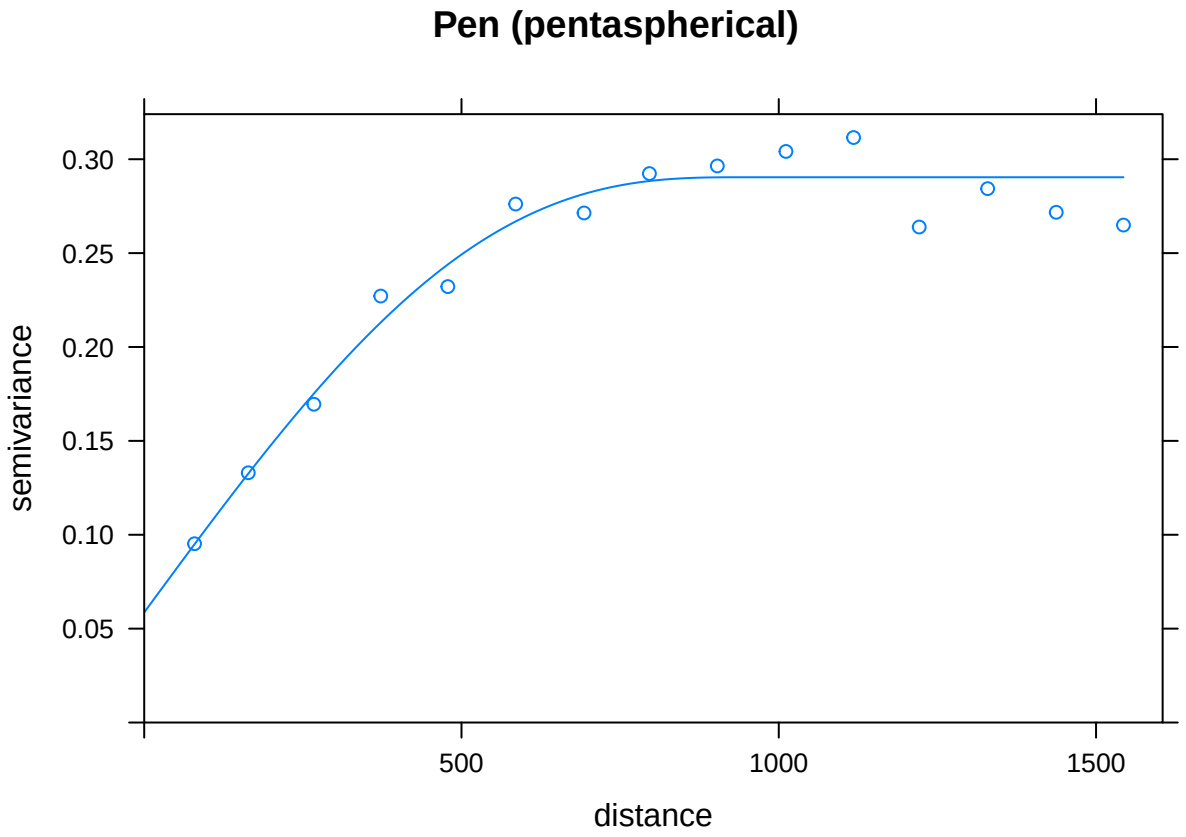
```
##    short                                      long        SSErr
## 1    Nug                                Nug (nugget)           NA
## 2    Exp                           Exp (exponential) 3.709119e-06
## 3    Sph                             Sph (spherical) 2.266095e-06
## 4    Gau                              Gau (gaussian) 2.951113e-06
## 5    Exc        Exclass (Exponential class/stable) 9.800221e-06
## 6    Mat                                Mat (Matern) 3.709119e-06
## 7    Ste Mat (Matern, M. Stein's parameterization) 3.709119e-06
## 8    Cir                              Cir (circular) 2.436197e-06
## 9    Lin                                Lin (linear) 3.102196e-06
## 10   Bes                                Bes (bessel) 2.645632e-06
## 11   Pen                       Pen (pentaspherical) 2.171773e-06
## 12   Per                              Per (periodic) 1.953203e-04
## 13   Wav                                  Wav (wave) 4.001240e-05
## 14   Hol                                  Hol (hole) 5.420405e-04
## 15   Log                           Log (logarithmic) 8.887894e-02
## 16   Pow                                Pow (power)           NA
## 17   Spl                              Spl (spline) 2.707089e+09
## 18   Leg                             Leg (Legendre) 1.532755e-02
## 19   Err                    Err (Measurement error)           NA
## 20   Int                             Int (Intercept)           NA
```

10 Mejor modelo

```
(mejor.modelo = modelos[which.min(modelos$SSErr),])
```

```
##    short                 long        SSErr
## 11   Pen Pen (pentaspherical) 2.171773e-06
```

```
lCu.fit = fit.variogram(lCu.vgm,
              model = vgm(0.25, mejor.modelo$short, 900, 0.05))
plot(lCu.vgm, lCu.fit, main=as.character(mejor.modelo$long))
```



11 Kriging Ordinario

```
lCu.kriged = krige(log(copper)~1, meuse, meuse.grid, model = lCu.fit)
```

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

```
summary(lCu.kriged)
```

```
## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min    max
## x 178460 181540
## y 329620 333740
## Is projected: NA
## proj4string : [NA]
## Number of points: 3103
## Grid attributes:
##   cellcentre.offset cellsize cells.dim
## x            178460       40        78
## y            329620       40       104
## Data attributes:
##    var1.pred        var1.var
##  Min.   :2.775   Min.   :0.08361
##  1st Qu.:3.191   1st Qu.:0.10879
##  Median :3.367   Median :0.12083
```

```
##  Mean   :3.447   Mean   :0.13192
##  3rd Qu.:3.637   3rd Qu.:0.14536
##  Max.   :4.625   Max.   :0.26575
```

```
names(lCu.kriged)
```

```
## [1] "var1.pred" "var1.var"
```

```
dim(lCu.kriged)
```

```
## [1] 3103    2
```

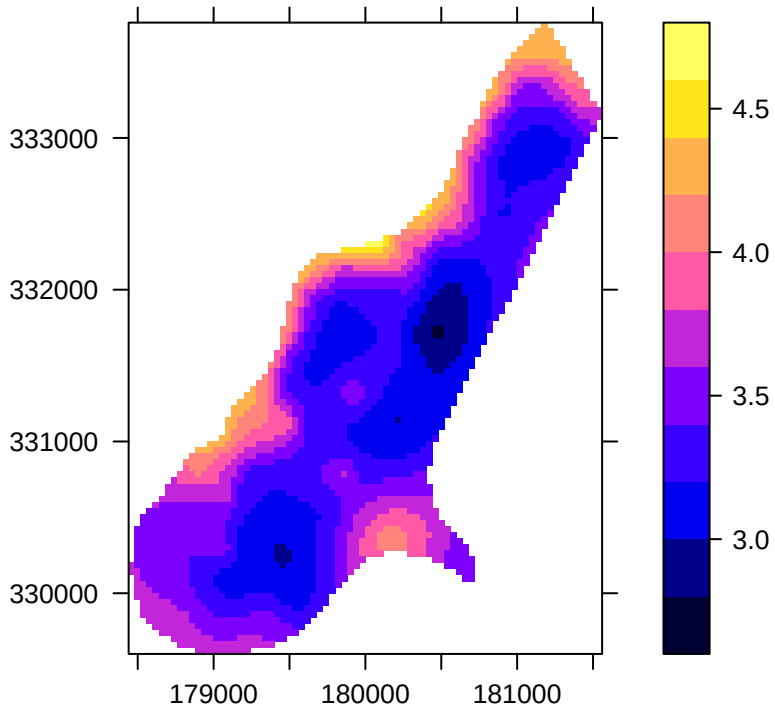
```
lCu.kriged$var1.pred[1:5]   # Predicción en los cinco primeros casos
```

```
## [1] 4.200836 4.254413 4.238081 4.220696 4.303937
```

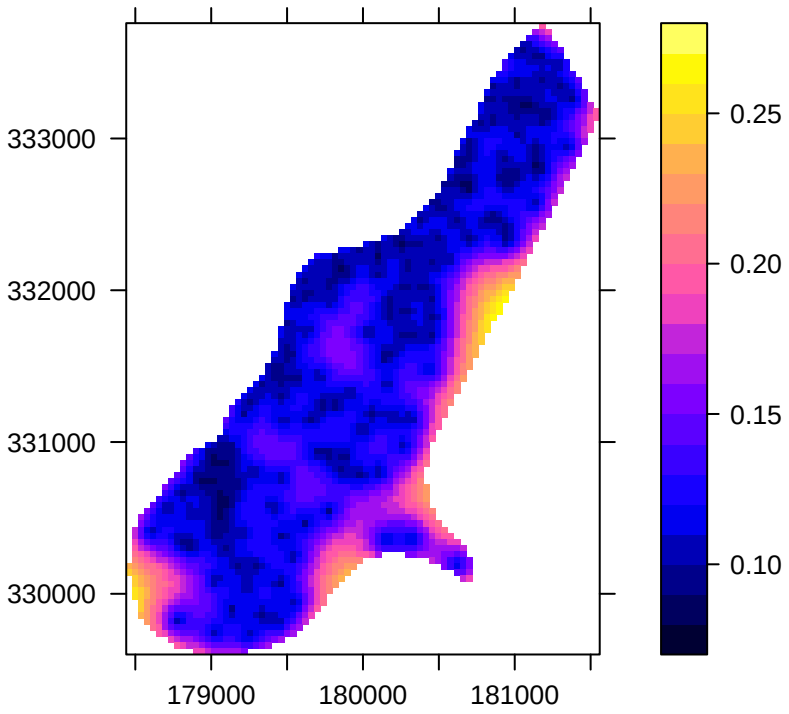
```
lCu.kriged$var1.var[1:5]    # Varianza de la Predicción en los cinco primeros casos
```

```
## [1] 0.1907213 0.1619264 0.1707465 0.1804112 0.1300948
```

```
spplot(lCu.kriged["var1.pred"], pretty=T, col.regions=bpy.colors(64),
       main="Predicción. Log(Cobre).",
       scales=list(draw=T))
```



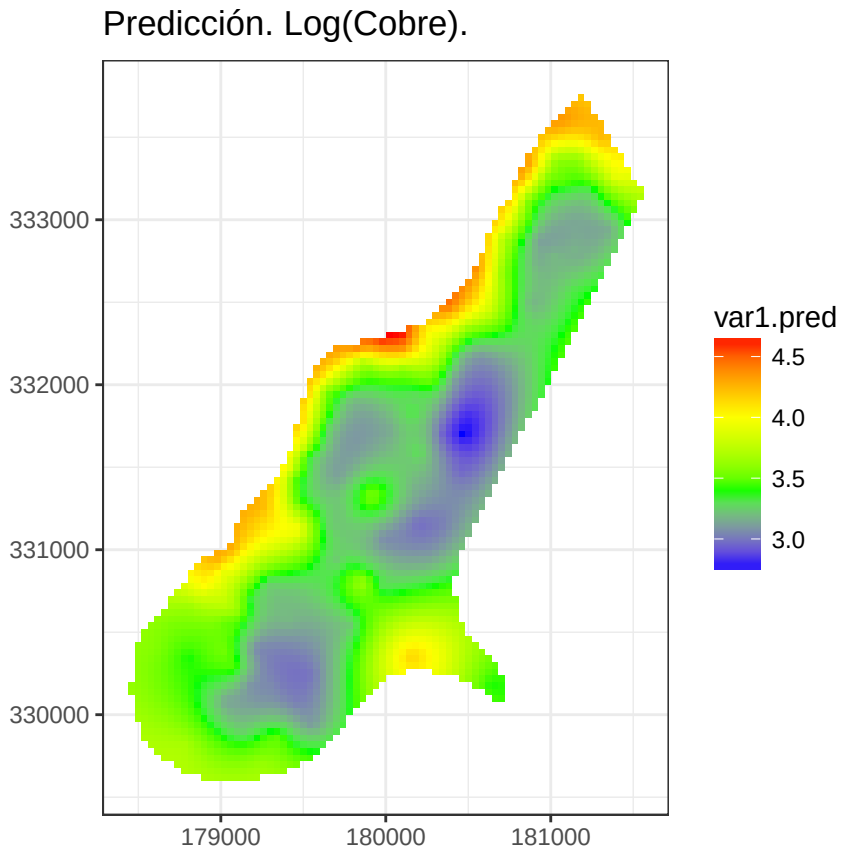

```
spplot(lCu.kriged["var1.var"], pretty=T, col.regions=bpy.colors(64),
       main="Varianza de Predicción. Log(Cobre).",
       scales=list(draw=T))
```



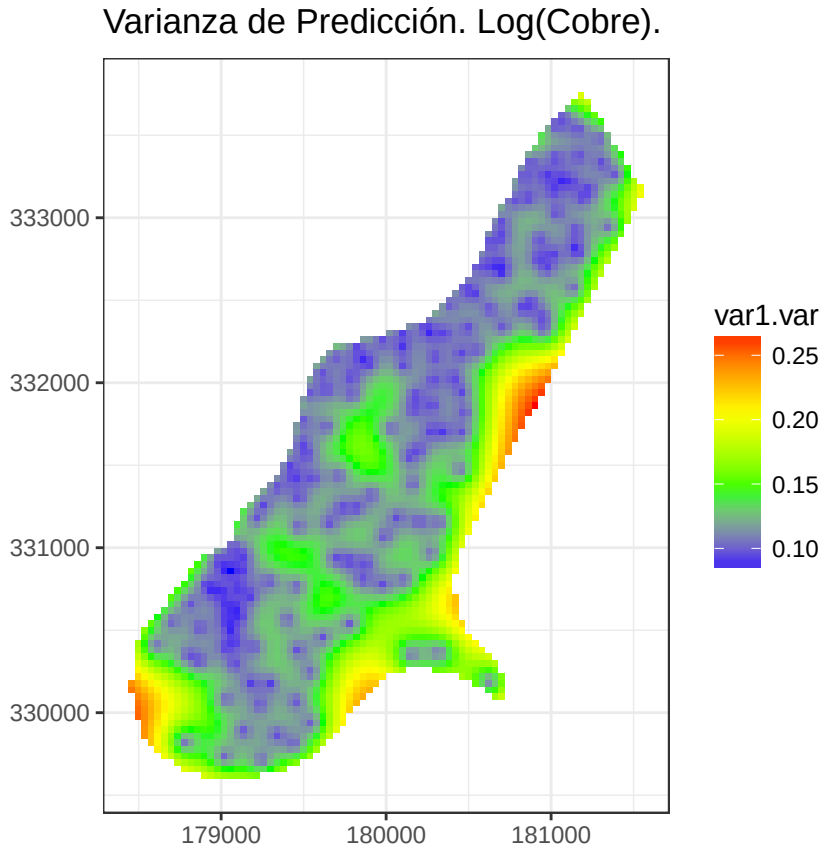

```
library(ggplot2)
df.lCu.kriged = as.data.frame(lCu.kriged)
head(df.lCu.kriged)
```

```
##        x      y var1.pred  var1.var
## 1 181180 333740  4.200836 0.1907213
## 2 181140 333700  4.254413 0.1619264
## 3 181180 333700  4.238081 0.1707465
## 4 181220 333700  4.220696 0.1804112
## 5 181100 333660  4.303937 0.1300948
## 6 181140 333660  4.283831 0.1399075
```

```
ggplot(df.lCu.kriged, aes(x,y,fill=var1.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Predicción. Log(Cobre).')
```




```
ggplot(df.lCu.kriged, aes(x,y,fill=var1.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Varianza de Predicción. Log(Cobre).')
```



12 Kriging Universal

12.1 Ajuste lineal sobre coordenadas y sus residuos

```
# Sin tranformación
summary(lm(formula=copper ~ coordinates(meuse), data=meuse))
```

```
##
## Call:
## lm(formula = copper ~ coordinates(meuse), data = meuse)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -31.565 -15.170  -5.975  10.384  75.887
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.508e+03  5.379e+02  -4.663 6.78e-06 ***
## coordinates(meuse)x -2.571e-02  4.351e-03  -5.909 2.18e-08 ***
## coordinates(meuse)y  2.164e-02  3.098e-03   6.984 8.38e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.74 on 152 degrees of freedom
```

```
## Multiple R-squared:  0.243,  Adjusted R-squared:  0.2331
## F-statistic:  24.4 on 2 and 152 DF,  p-value: 6.456e-10
```

```
# Log
lCu<-log(meuse$copper)
summary(lm(formula=lCu ~ coordinates(meuse), data=meuse))
```

```
##
## Call:
## lm(formula = lCu ~ coordinates(meuse), data = meuse)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7666 -0.3232 -0.1065  0.3308  1.4834
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -4.927e+01  1.138e+01  -4.331 2.68e-05 ***
## coordinates(meuse)x -5.998e-04  9.203e-05  -6.518 9.92e-10 ***
## coordinates(meuse)y  4.849e-04  6.553e-05   7.399 8.67e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4386 on 152 degrees of freedom
## Multiple R-squared:  0.2652, Adjusted R-squared:  0.2555
## F-statistic: 27.43 on 2 and 152 DF,  p-value: 6.732e-11
```

12.2 Variograma de residuos y comparación

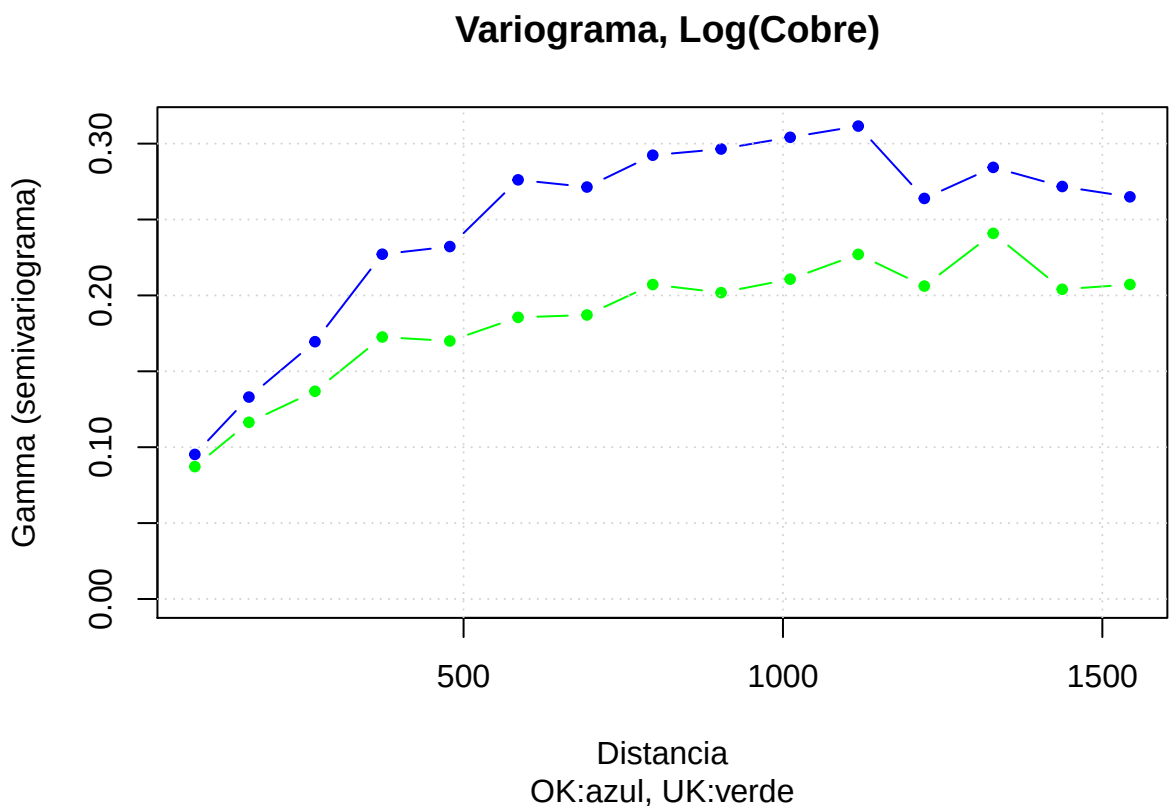
```
(lCu.res.vgm = variogram(log(copper)~x+y, meuse))
```

```
##     np       dist      gamma dir.hor dir.ver   id
## 1   57   79.29244 0.08719362       0       0 var1
## 2  299  163.97367 0.11641112       0       0 var1
## 3  419  267.36483 0.13683373       0       0 var1
## 4  457  372.73542 0.17257971       0       0 var1
## 5  547  478.47670 0.16997099       0       0 var1
## 6  533  585.34058 0.18554479       0       0 var1
## 7  574  693.14526 0.18709522       0       0 var1
## 8  564  796.18365 0.20714739       0       0 var1
## 9  589  903.14650 0.20182487       0       0 var1
## 10 543 1011.29177 0.21069077       0       0 var1
## 11 500 1117.86235 0.22705462       0       0 var1
## 12 477 1221.32810 0.20614711       0       0 var1
## 13 452 1329.16407 0.24085510       0       0 var1
## 14 457 1437.25620 0.20403254       0       0 var1
## 15 415 1543.20248 0.20717259       0       0 var1
```

```
(lCu.all.vgm <- data.frame(np = lCu.vgm$np,
                           dist = lCu.vgm$dist,
                           gamma.ok=lCu.vgm$gamma,
                           gamma.uk=lCu.res.vgm$gamma,
                           gamma.dif = lCu.vgm$gamma - lCu.res.vgm$gamma ))
```

```
##      np       dist   gamma.ok   gamma.uk   gamma.dif
## 1    57   79.29244 0.09522828 0.08719362 0.008034657
## 2   299  163.97367 0.13301890 0.11641112 0.016607784
## 3   419  267.36483 0.16949808 0.13683373 0.032664342
## 4   457  372.73542 0.22712857 0.17257971 0.054548855
## 5   547  478.47670 0.23217426 0.16997099 0.062203271
## 6   533  585.34058 0.27613309 0.18554479 0.090588300
## 7   574  693.14526 0.27139217 0.18709522 0.084296955
## 8   564  796.18365 0.29236481 0.20714739 0.085217422
## 9   589  903.14650 0.29642987 0.20182487 0.094604992
## 10 543 1011.29177 0.30415935 0.21069077 0.093468577
## 11 500 1117.86235 0.31154663 0.22705462 0.084492012
## 12 477 1221.32810 0.26389361 0.20614711 0.057746499
## 13 452 1329.16407 0.28434482 0.24085510 0.043489719
## 14 457 1437.25620 0.27174256 0.20403254 0.067710018
## 15 415 1543.20248 0.26493408 0.20717259 0.057761485
```

```r
plot(lCu.all.vgm$gamma.ok ~ lCu.all.vgm$dist, pch=20, col="blue",
     type="b", xlab="Distancia", ylab="Gamma (semivariograma)",
     ylim=c(0,max(lCu.all.vgm$gamma.ok, lCu.all.vgm$gamma.uk)),
     main = " Variograma, Log(Cobre)", sub="OK:azul, UK:verde")
points(lCu.all.vgm$gamma.uk  ~ lCu.all.vgm$dist, pch=20, col="green",
       type="b")
grid()
```



12.3 Ajuste de modelos teóricos

```
modelos = data.frame(modelos,'SSErr_res'=NA)
for (i in c(2:15,17,18)) { # Se han excluido manualmente los que han dado error
    modelos$SSErr_res[i] = attributes(fit.variogram(
      lCu.res.vgm, model=vgm(0.20, modelos$short[i], 900, 0.05)))$SSErr
}
```

```
## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.res.vgm, model = vgm(0.2, modelos$short[i], :
## No convergence after 200 iterations: try different initial values?
```

```
## Warning in fit.variogram(lCu.res.vgm, model = vgm(0.2, modelos$short[i], :
## singular model in variogram fit
```

```
## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges
## = fit.ranges, : No convergence after 200 iterations: try different initial
## values?
```

```
## Warning in fit.variogram(lCu.res.vgm, model = vgm(0.2, modelos$short[i], :
## singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.res.vgm, model = vgm(0.2, modelos$short[i], :
## singular model in variogram fit
```

```
modelos
```

```
##    short                                      long        SSErr
## 1    Nug                           Nug (nugget)           NA
## 2    Exp                       Exp (exponential) 3.709119e-06
## 3    Sph                         Sph (spherical) 2.266095e-06
## 4    Gau                          Gau (gaussian) 2.951113e-06
## 5    Exc       Exclass (Exponential class/stable) 9.800221e-06
## 6    Mat                            Mat (Matern) 3.709119e-06
## 7    Ste Mat (Matern, M. Stein's parameterization) 3.709119e-06
## 8    Cir                           Cir (circular) 2.436197e-06
## 9    Lin                             Lin (linear) 3.102196e-06
## 10   Bes                             Bes (bessel) 2.645632e-06
## 11   Pen                   Pen (pentaspherical) 2.171773e-06
## 12   Per                         Per (periodic) 1.953203e-04
## 13   Wav                             Wav (wave) 4.001240e-05
## 14   Hol                             Hol (hole) 5.420405e-04
## 15   Log                     Log (logarithmic) 8.887894e-02
## 16   Pow                           Pow (power)           NA
## 17   Spl                           Spl (spline) 2.707089e+09
## 18   Leg                         Leg (Legendre) 1.532755e-02
## 19   Err                 Err (Measurement error)           NA
## 20   Int                          Int (Intercept)           NA
##       SSErr_res
## 1            NA
## 2  1.107682e-06
## 3  1.976535e-06
## 4  2.626598e-06
## 5  1.369745e-06
## 6  1.107682e-06
```

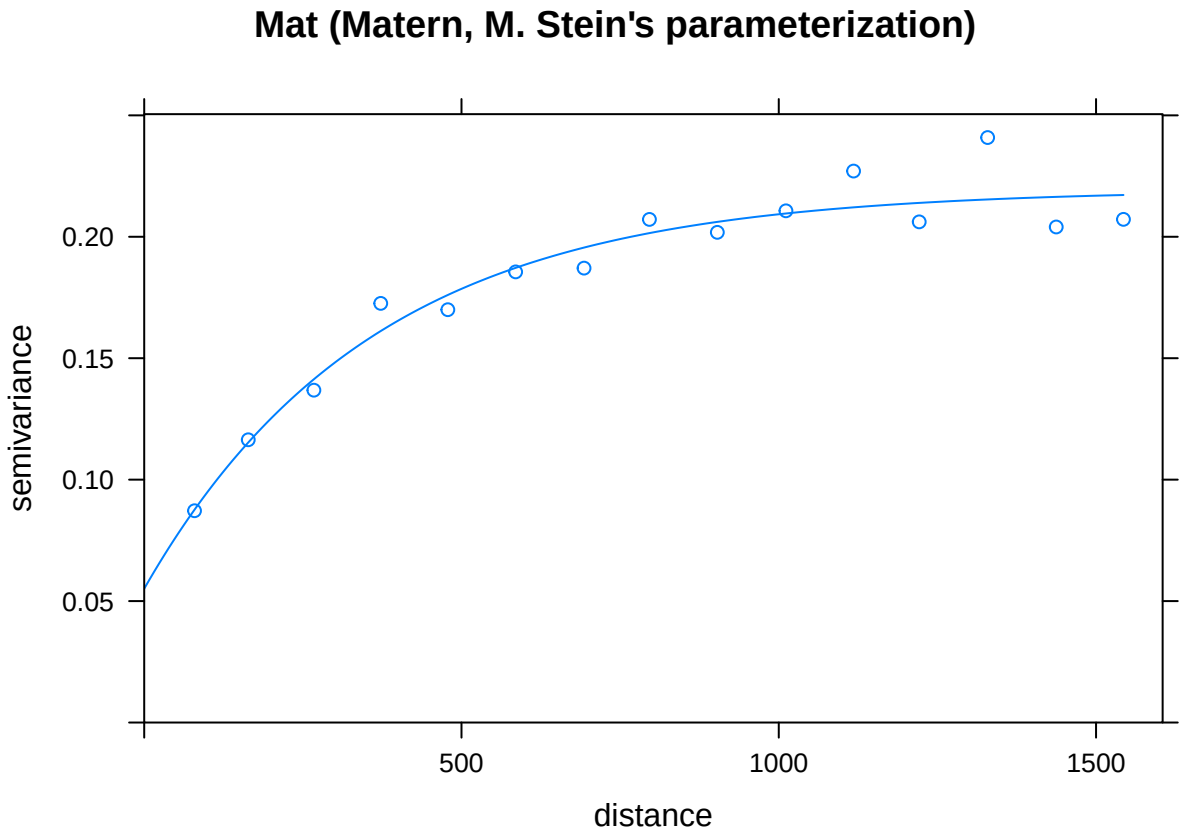
```
## 7  1.107682e-06
## 8  2.252892e-06
## 9  3.068402e-06
## 10 1.287015e-06
## 11 1.724752e-06
## 12 6.934794e-05
## 13 1.357247e-05
## 14 8.708338e-05
## 15 6.431726e-04
## 16           NA
## 17 1.732537e+09
## 18 1.016827e-02
## 19           NA
## 20           NA
```

12.4 Mejor modelo

```
(mejor.modelo = modelos[which.min(modelos$SSErr_res),])
```

```
##   short                                       long        SSErr
## 7   Ste Mat (Matern, M. Stein's parameterization) 3.709119e-06
##      SSErr_res
## 7 1.107682e-06
```

```
lCu.res.fit = fit.variogram(lCu.res.vgm,
                model = vgm(0.20, mejor.modelo$short, 900, 0.05))
plot(lCu.res.vgm, lCu.res.fit, main=as.character(mejor.modelo$long))
```



12.5 Predicciones

```
lCu.Ukriged = krige(log(copper)~x+y,
                    meuse, meuse.grid, model = lCu.res.fit)
```

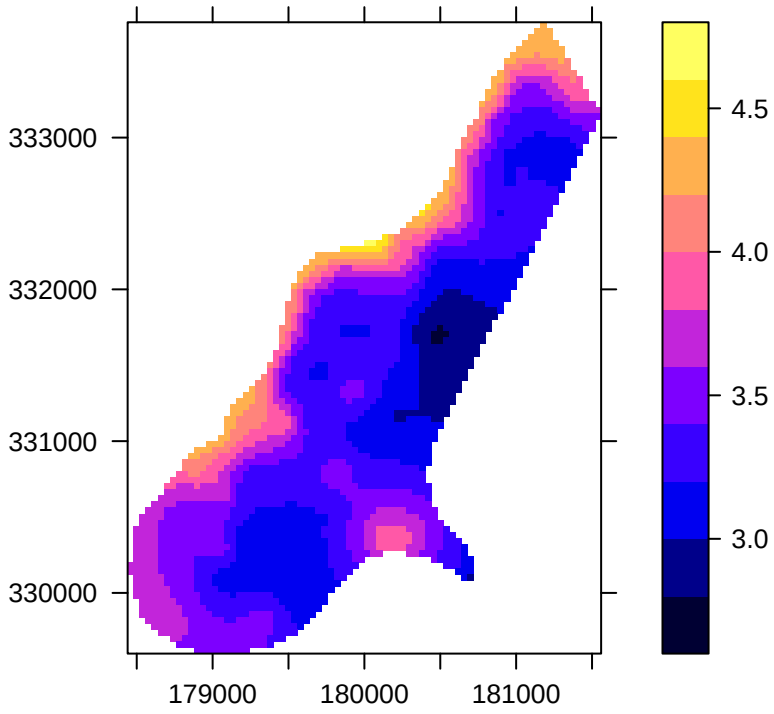
```
## [using universal kriging]
```

```
summary(lCu.Ukriged)
```

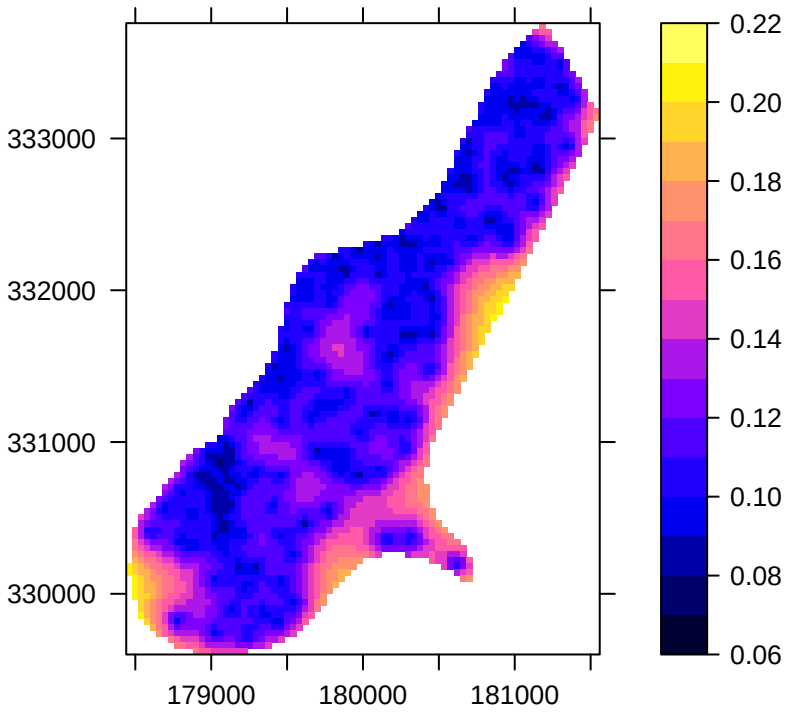
```
## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min    max
## x 178460 181540
## y 329620 333740
## Is projected: NA
## proj4string : [NA]
## Number of points: 3103
## Grid attributes:
##   cellcentre.offset cellsize cells.dim
## x            178460       40        78
## y            329620       40       104
## Data attributes:
##    var1.pred        var1.var
##  Min.   :2.772   Min.   :0.07894
##  1st Qu.:3.191   1st Qu.:0.10212
```

```
##  Median :3.335   Median :0.11256
##  Mean   :3.424   Mean   :0.11955
##  3rd Qu.:3.610   3rd Qu.:0.13166
##  Max.   :4.614   Max.   :0.20703
```

```
spplot(lCu.Ukriged["var1.pred"], pretty=T, col.regions=bpy.colors(64),
       main="Predicción UK. Log(Cobre).",
       scales=list(draw=T))
```



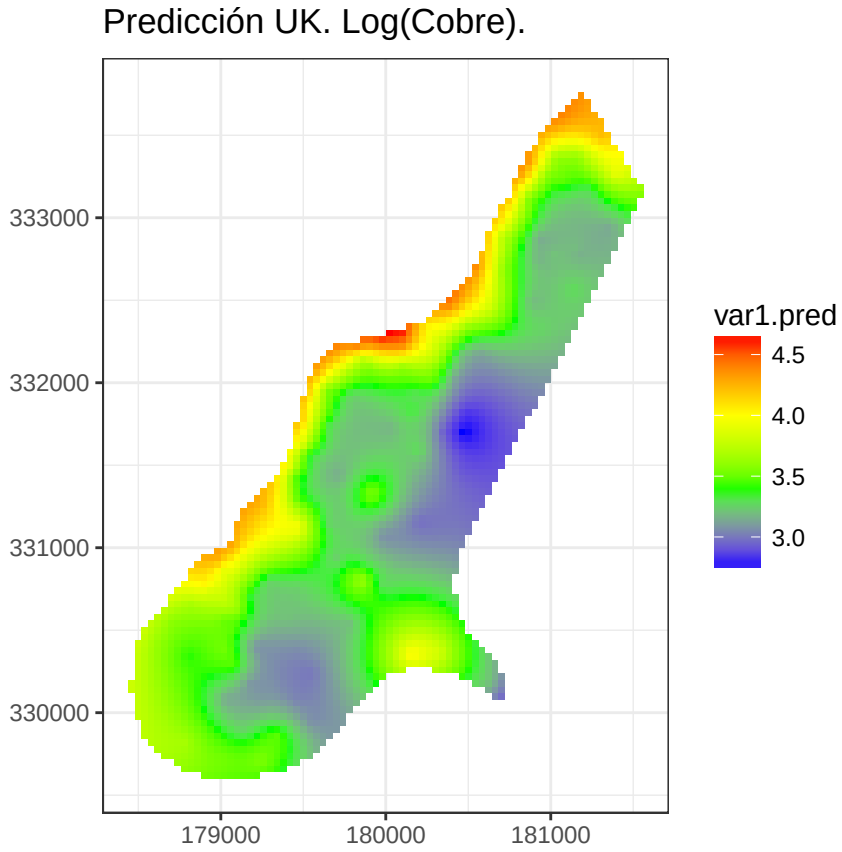

```
spplot(lCu.Ukriged["var1.var"], pretty=T, col.regions=bpy.colors(64),
       main="Varianza de Predicción UK. Log(Cobre).",
       scales=list(draw=T))
```



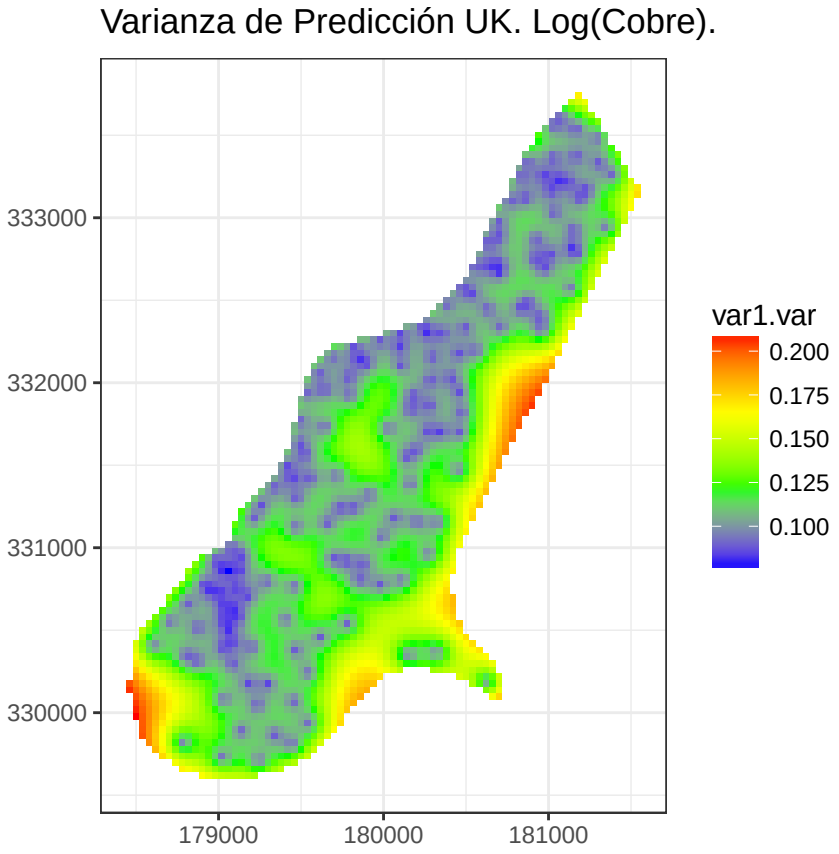

```
df.lCu.Ukriged = as.data.frame(lCu.Ukriged)
head(df.lCu.Ukriged)
```

```
##        x      y var1.pred  var1.var
## 1 181180 333740  4.353439 0.1674917
## 2 181140 333700  4.370373 0.1455395
## 3 181180 333700  4.344553 0.1521081
## 4 181220 333700  4.319174 0.1591605
## 5 181100 333660  4.383838 0.1197542
## 6 181140 333660  4.355609 0.1278914
```

```
ggplot(df.lCu.Ukriged, aes(x,y,fill=var1.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Predicción UK. Log(Cobre).')
```




```
ggplot(df.lCu.Ukriged, aes(x,y,fill=var1.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Varianza de Predicción UK. Log(Cobre).')
```

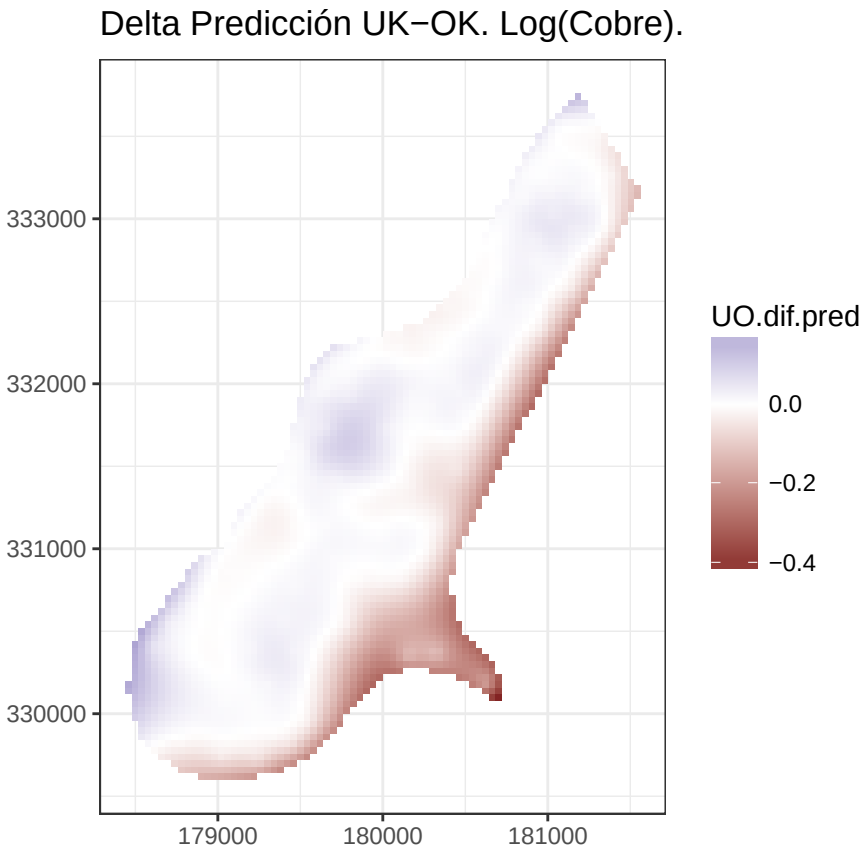


12.6 Comparativa

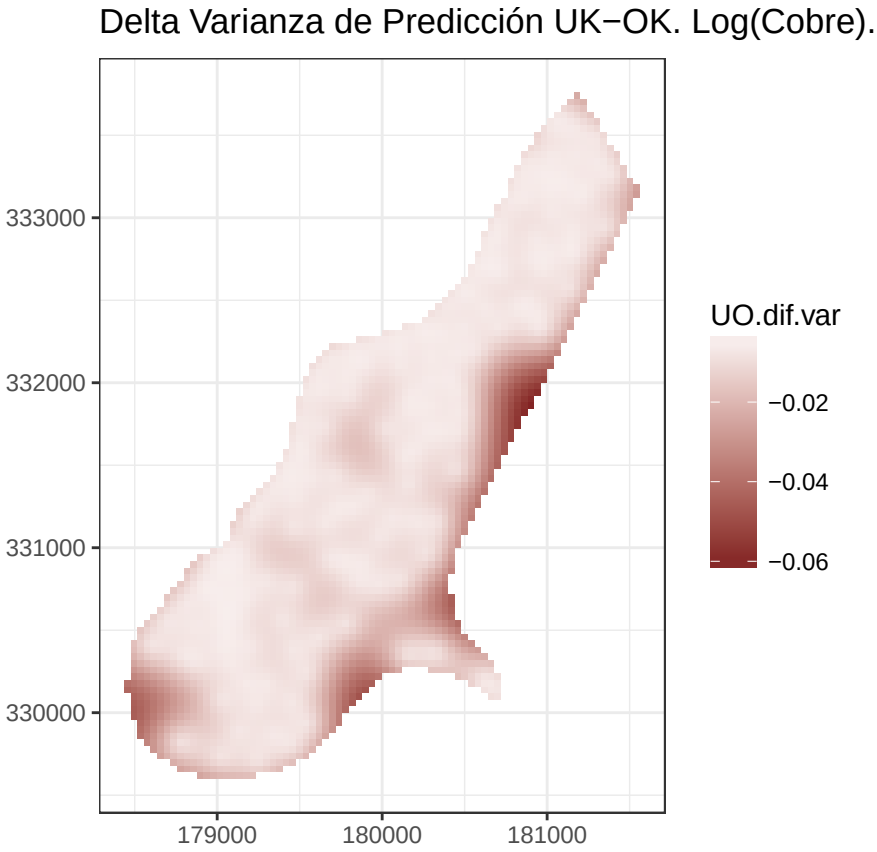
```
df.lCu.preds = data.frame(
  df.lCu.kriged,
  U.pred = lCu.Ukriged$var1.pred, U.var = lCu.Ukriged$var1.var,
  UO.dif.pred = lCu.Ukriged$var1.pred - lCu.kriged$var1.pred,
  UO.dif.var = lCu.Ukriged$var1.var - lCu.kriged$var1.var
)

summary(df.lCu.preds)
```

```
##        x                y            var1.pred        var1.var
##  Min.   :178460   Min.   :329620   Min.   :2.775   Min.   :0.08361
##  1st Qu.:179420   1st Qu.:330460   1st Qu.:3.191   1st Qu.:0.10879
##  Median :179980   Median :331220   Median :3.367   Median :0.12083
##  Mean   :179985   Mean   :331348   Mean   :3.447   Mean   :0.13192
##  3rd Qu.:180580   3rd Qu.:332140   3rd Qu.:3.637   3rd Qu.:0.14536
##  Max.   :181540   Max.   :333740   Max.   :4.625   Max.   :0.26575
##      U.pred           U.var          UO.dif.pred          UO.dif.var
##  Min.   :2.772   Min.   :0.07894   Min.   :-0.444217   Min.   :-0.061627
##  1st Qu.:3.191   1st Qu.:0.10212   1st Qu.:-0.034734   1st Qu.:-0.013732
##  Median :3.335   Median :0.11256   Median : 0.003102   Median :-0.008372
##  Mean   :3.424   Mean   :0.11955   Mean   :-0.023218   Mean   :-0.012369
##  3rd Qu.:3.610   3rd Qu.:0.13166   3rd Qu.: 0.019132   3rd Qu.:-0.006629
##  Max.   :4.614   Max.   :0.20703   Max.   : 0.193378   Max.   :-0.004670
```

```
ggplot(df.lCu.preds, aes(x,y,fill=UO.dif.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradient2() +
  labs(x=NULL,y=NULL,
       title='Delta Predicción UK-OK. Log(Cobre).')
```




```
ggplot(df.lCu.preds, aes(x,y,fill=UO.dif.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradient2() +
  labs(x=NULL,y=NULL,
       title='Delta Varianza de Predicción UK-OK. Log(Cobre).')
```

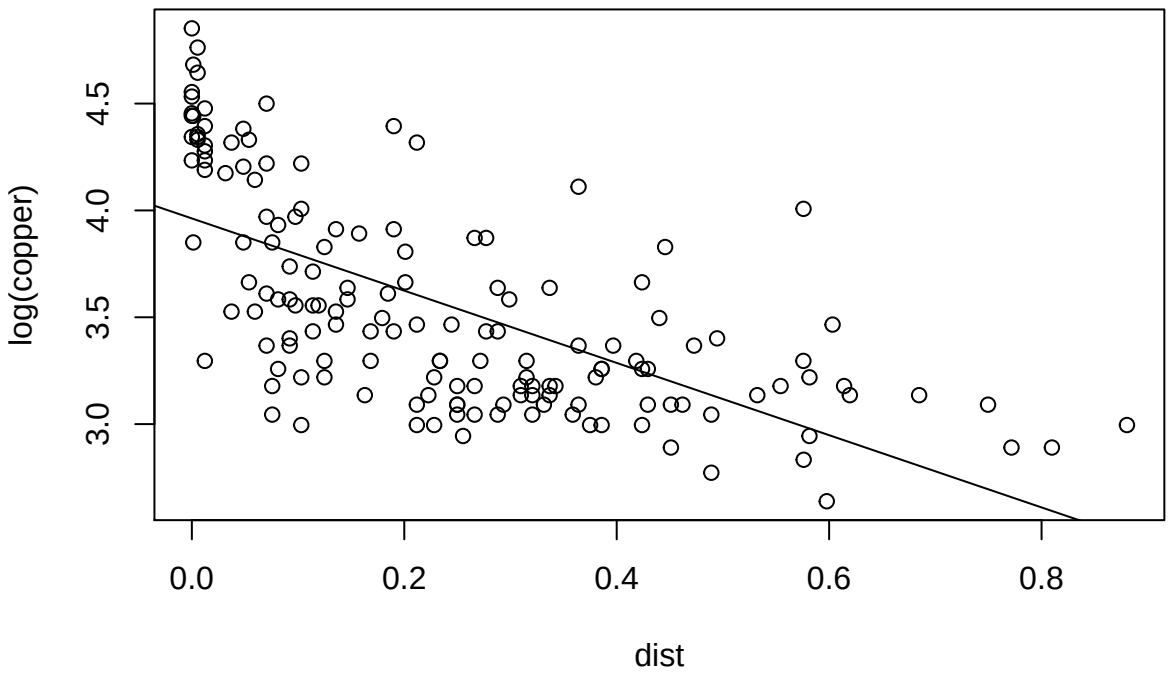


Conclusión: No existen grandes diferencias pero Universal Kriging es preferible a Kriging Ordinario en este caso por mostrar menor varianza de las predicciones.

13 Kriging Deriva Externa

13.1 Ajuste lineal

```
plot(log(copper)~ dist, meuse)
abline(lm(formula=log(copper) ~ dist, data=meuse))
```




```
summary(lm(formula=log(copper) ~ dist, data=meuse))
```

```
##
## Call:
## lm(formula = log(copper) ~ dist, data = meuse)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.79260 -0.29425 -0.05632  0.28662  1.01837
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.96251    0.04864   81.47   <2e-16 ***
## dist        -1.69055    0.15662  -10.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3843 on 153 degrees of freedom
## Multiple R-squared:  0.4323, Adjusted R-squared:  0.4286
## F-statistic: 116.5 on 1 and 153 DF,  p-value: < 2.2e-16
```

```
# summary(lm(formula=log(copper) ~ coordinates(meuse)+dist, data=meuse))
```

13.2 Variograma de residuos

```
(lCu.rdist.vgm = variogram(log(copper)~dist, meuse))
```

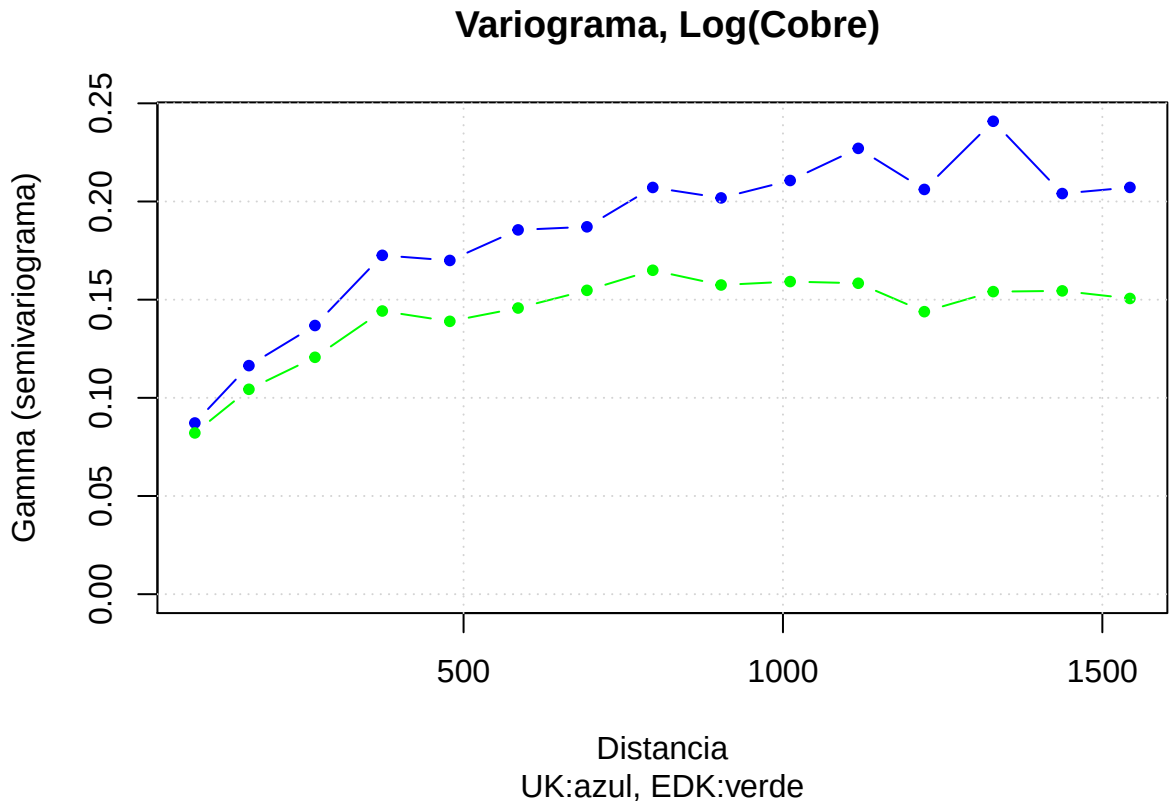
```
##     np       dist      gamma dir.hor dir.ver   id
## 1   57   79.29244 0.08212415       0       0 var1
## 2  299  163.97367 0.10434468       0       0 var1
## 3  419  267.36483 0.12064817       0       0 var1
## 4  457  372.73542 0.14421991       0       0 var1
## 5  547  478.47670 0.13896273       0       0 var1
## 6  533  585.34058 0.14574394       0       0 var1
## 7  574  693.14526 0.15474880       0       0 var1
## 8  564  796.18365 0.16497612       0       0 var1
## 9  589  903.14650 0.15746204       0       0 var1
## 10 543 1011.29177 0.15921954       0       0 var1
## 11 500 1117.86235 0.15836408       0       0 var1
## 12 477 1221.32810 0.14388508       0       0 var1
## 13 452 1329.16407 0.15409336       0       0 var1
## 14 457 1437.25620 0.15447101       0       0 var1
## 15 415 1543.20248 0.15061042       0       0 var1
```

```
(lCu.all.vgm = data.frame(
  lCu.all.vgm,
  gamma.edk = lCu.rdist.vgm$gamma,
  gamma.dif.edk = lCu.res.vgm$gamma - lCu.rdist.vgm$gamma
  )
)
```

```
##     np       dist   gamma.ok   gamma.uk   gamma.dif  gamma.edk
## 1   57   79.29244 0.09522828 0.08719362 0.008034657 0.08212415
## 2  299  163.97367 0.13301890 0.11641112 0.016607784 0.10434468
## 3  419  267.36483 0.16949808 0.13683373 0.032664342 0.12064817
## 4  457  372.73542 0.22712857 0.17257971 0.054548855 0.14421991
## 5  547  478.47670 0.23217426 0.16997099 0.062203271 0.13896273
## 6  533  585.34058 0.27613309 0.18554479 0.090588300 0.14574394
## 7  574  693.14526 0.27139217 0.18709522 0.084296955 0.15474880
## 8  564  796.18365 0.29236481 0.20714739 0.085217422 0.16497612
## 9  589  903.14650 0.29642987 0.20182487 0.094604992 0.15746204
## 10 543 1011.29177 0.30415935 0.21069077 0.093468577 0.15921954
## 11 500 1117.86235 0.31154663 0.22705462 0.084492012 0.15836408
## 12 477 1221.32810 0.26389361 0.20614711 0.057746499 0.14388508
## 13 452 1329.16407 0.28434482 0.24085510 0.043489719 0.15409336
## 14 457 1437.25620 0.27174256 0.20403254 0.067710018 0.15447101
## 15 415 1543.20248 0.26493408 0.20717259 0.057761485 0.15061042
##    gamma.dif.edk
## 1    0.005069476
## 2    0.012066431
## 3    0.016185560
## 4    0.028359796
## 5    0.031008252
## 6    0.039800849
## 7    0.032346422
## 8    0.042171271
## 9    0.044362830
```

```
## 10   0.051471236
## 11   0.068690543
## 12   0.062262031
## 13   0.086761747
## 14   0.049561535
## 15   0.056562173
```

```
plot(lCu.all.vgm$gamma.uk ~ lCu.all.vgm$dist, pch=20, col="blue",
     type="b", xlab="Distancia", ylab="Gamma (semivariograma)",
     ylim=c(0,max(lCu.all.vgm$gamma.uk, lCu.all.vgm$gamma.edk)),
     main = " Variograma, Log(Cobre)", sub="UK:azul, EDK:verde")
points(lCu.all.vgm$gamma.edk  ~ lCu.all.vgm$dist, pch=20, col="green",
       type="b")
grid()
```



13.3 Ajuste de modelos teóricos

```
modelos = data.frame(modelos,'SSErr_rdist'=NA)
for (i in c(2:15,17,18)) { # Se han excluido manualmente los que han dado error
    modelos$SSErr_rdist[i] = attributes(fit.variogram(
      lCu.rdist.vgm, model=vgm(0.15, modelos$short[i], 500, 0.05)))$SSErr
}
```

```
## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.rdist.vgm, model = vgm(0.15, modelos
## $short[i], : No convergence after 200 iterations: try different initial
## values?
```

```
## Warning in fit.variogram(lCu.rdist.vgm, model = vgm(0.15, modelos
## $short[i], : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.rdist.vgm, model = vgm(0.15, modelos
## $short[i], : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.rdist.vgm, model = vgm(0.15, modelos
## $short[i], : singular model in variogram fit
```

```
modelos
```

```
##    short                                       long        SSErr
## 1    Nug                                Nug (nugget)           NA
## 2    Exp                            Exp (exponential) 3.709119e-06
## 3    Sph                              Sph (spherical) 2.266095e-06
## 4    Gau                               Gau (gaussian) 2.951113e-06
## 5    Exc        Exclass (Exponential class/stable) 9.800221e-06
## 6    Mat                                Mat (Matern) 3.709119e-06
## 7    Ste Mat (Matern, M. Stein's parameterization) 3.709119e-06
## 8    Cir                               Cir (circular) 2.436197e-06
## 9    Lin                                 Lin (linear) 3.102196e-06
## 10   Bes                                 Bes (bessel) 2.645632e-06
## 11   Pen                       Pen (pentaspherical) 2.171773e-06
## 12   Per                             Per (periodic) 1.953203e-04
## 13   Wav                                 Wav (wave) 4.001240e-05
## 14   Hol                                 Hol (hole) 5.420405e-04
## 15   Log                            Log (logarithmic) 8.887894e-02
## 16   Pow                                 Pow (power)           NA
## 17   Spl                               Spl (spline) 2.707089e+09
## 18   Leg                             Leg (Legendre) 1.532755e-02
## 19   Err                     Err (Measurement error)           NA
## 20   Int                               Int (Intercept)           NA
##       SSErr_res  SSErr_rdist
## 1            NA           NA
## 2  1.107682e-06 5.383024e-07
## 3  1.976535e-06 7.447525e-07
## 4  2.626598e-06 8.346376e-07
## 5  1.369745e-06 8.529689e-07
## 6  1.107682e-06 5.383024e-07
## 7  1.107682e-06 5.383024e-07
## 8  2.252892e-06 8.145562e-07
## 9  3.068402e-06 1.376126e-06
## 10 1.287015e-06 5.253159e-07
## 11 1.724752e-06 6.628005e-07
## 12 6.934794e-05 2.919926e-05
## 13 1.357247e-05 2.727819e-06
## 14 8.708338e-05 3.624011e-05
## 15 6.431726e-04 1.141330e-04
## 16           NA           NA
## 17 1.732537e+09 1.832742e+08
## 18 1.016827e-02 5.662254e-03
```

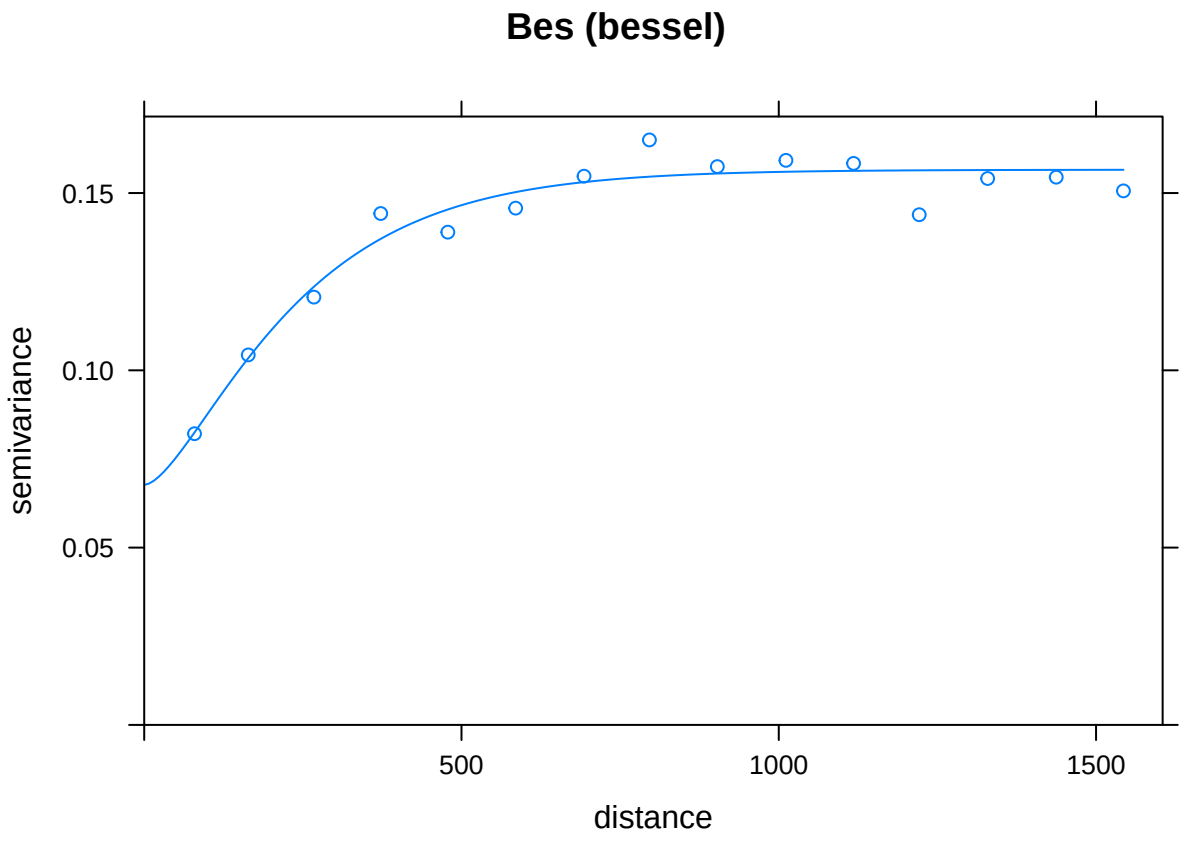
```
## 19           NA           NA
## 20           NA           NA
```

13.4 Mejor modelo

```
(mejor.modelo = modelos[which.min(modelos$SSErr_rdist),])
```

```
##    short         long        SSErr    SSErr_res  SSErr_rdist
## 10   Bes Bes (bessel) 2.645632e-06 1.287015e-06 5.253159e-07
```

```
lCu.rdist.fit = fit.variogram(lCu.rdist.vgm,
                model = vgm(0.15, mejor.modelo$short, 500, 0.05))
plot(lCu.rdist.vgm, lCu.rdist.fit, main=as.character(mejor.modelo$long))
```



13.5 Predicciones

```
lCu.EDkriged = krige(log(copper)~dist,meuse,
                     meuse.grid, model = lCu.rdist.fit)
```

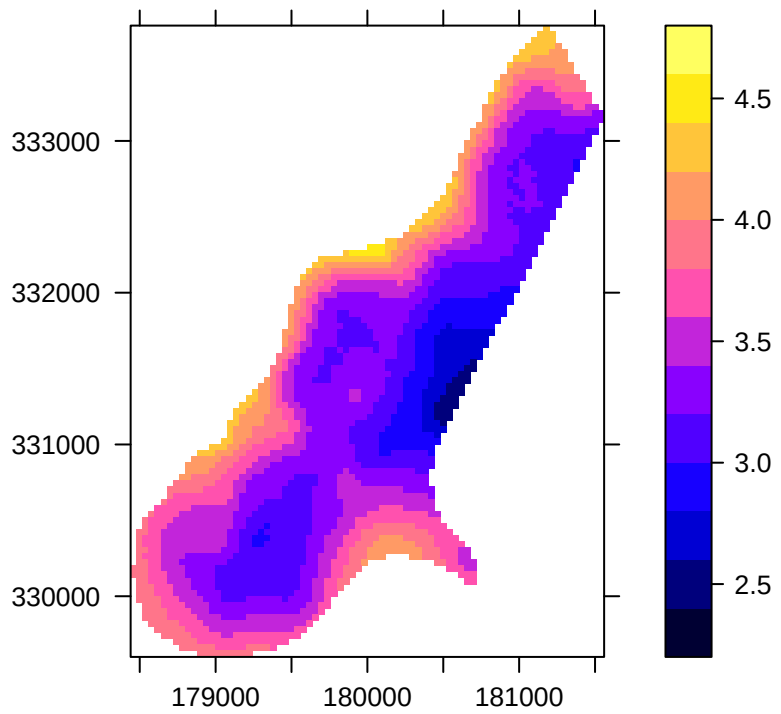
```
## [using universal kriging]
```

```
summary(lCu.EDkriged)
```

```
## Object of class SpatialPixelsDataFrame
## Coordinates:
```

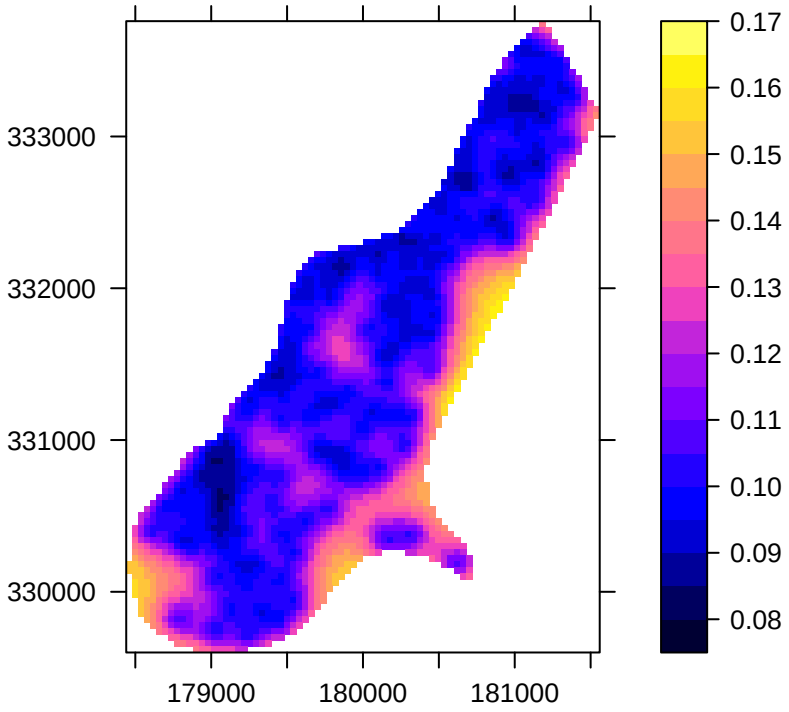
```
##      min    max
## x 178460 181540
## y 329620 333740
## Is projected: NA
## proj4string : [NA]
## Number of points: 3103
## Grid attributes:
##   cellcentre.offset cellsize cells.dim
## x            178460       40        78
## y            329620       40       104
## Data attributes:
##    var1.pred        var1.var
##  Min.   :2.434   Min.   :0.08370
##  1st Qu.:3.181   1st Qu.:0.09752
##  Median :3.362   Median :0.10458
##  Mean   :3.434   Mean   :0.10975
##  3rd Qu.:3.699   3rd Qu.:0.11863
##  Max.   :4.545   Max.   :0.16252
```

```
spplot(lCu.EDkriged["var1.pred"], pretty=T, col.regions=bpy.colors(64),
       main="Predicción EDK (Distancia al Río). Log(Cobre).",
       scales=list(draw=T))
```




```
spplot(lCu.EDkriged["var1.var"], pretty=T, col.regions=bpy.colors(64),
       main="Varianza de Predicción EDK (Distancia al Río). Log(Cobre).",
       scales=list(draw=T))
```

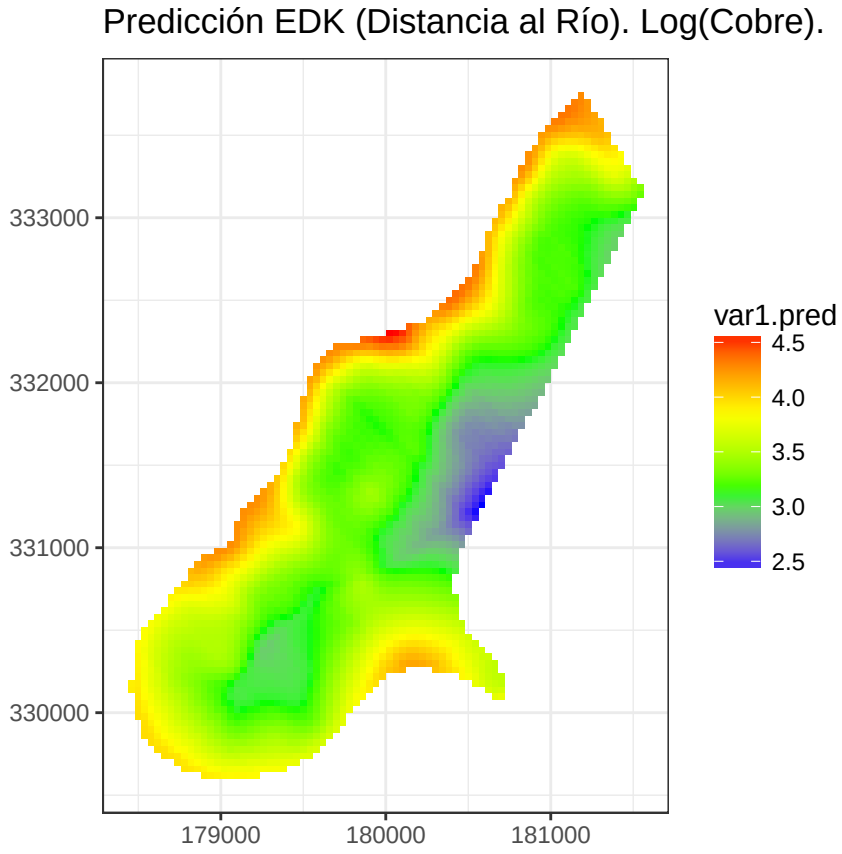
Varianza de Predicción EDK (Distancia al Río). Log(Cobre).




```
df.lCu.EDkriged = as.data.frame(lCu.EDkriged)
head(df.lCu.EDkriged)
```

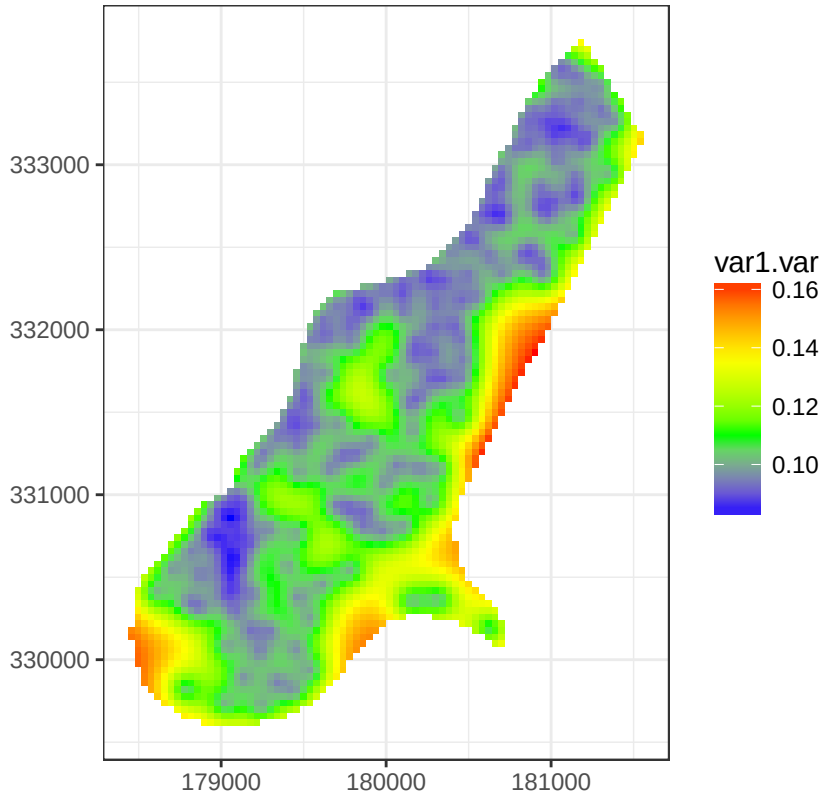
```
##        x      y var1.pred  var1.var
## 1 181180 333740  4.271363 0.1363428
## 2 181140 333700  4.305936 0.1237270
## 3 181180 333700  4.284158 0.1278679
## 4 181220 333700  4.225923 0.1318495
## 5 181100 333660  4.332084 0.1085873
## 6 181140 333660  4.313877 0.1131608
```

```
ggplot(df.lCu.EDkriged, aes(x,y,fill=var1.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Predicción EDK (Distancia al Río). Log(Cobre).')
```




```
ggplot(df.lCu.EDkriged, aes(x,y,fill=var1.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Varianza de Predicción EDK (Distancia al Río). Log(Cobre).')
```

Varianza de Predicción EDK (Distancia al Río). Log(Cobre).



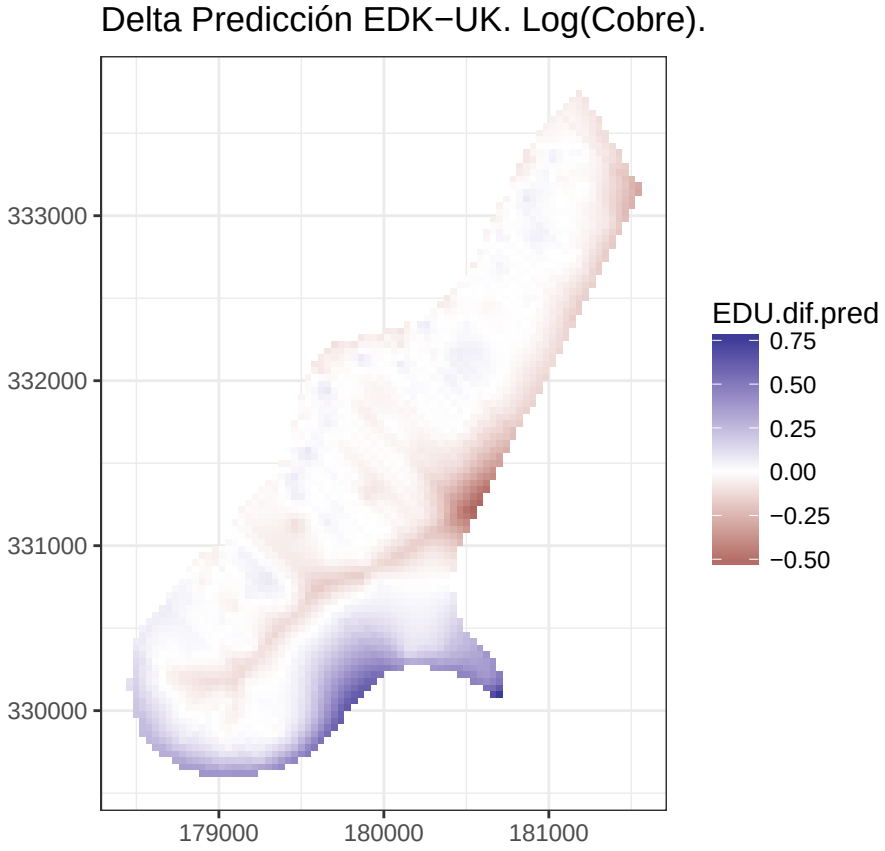
13.6 Comparativa

```
df.lCu.preds = data.frame(df.lCu.kriged,
  U.pred = lCu.Ukriged$var1.pred, U.var = lCu.Ukriged$var1.var,
  UO.dif.pred = lCu.Ukriged$var1.pred - lCu.kriged$var1.pred,
  UO.dif.var = lCu.Ukriged$var1.var - lCu.kriged$var1.var,
  ED.pred = lCu.EDkriged$var1.pred, ED.var = lCu.EDkriged$var1.var,
  EDU.dif.pred = lCu.EDkriged$var1.pred - lCu.Ukriged$var1.pred,
  EDU.dif.var = lCu.EDkriged$var1.var - lCu.Ukriged$var1.var
  )

summary(df.lCu.preds)
```

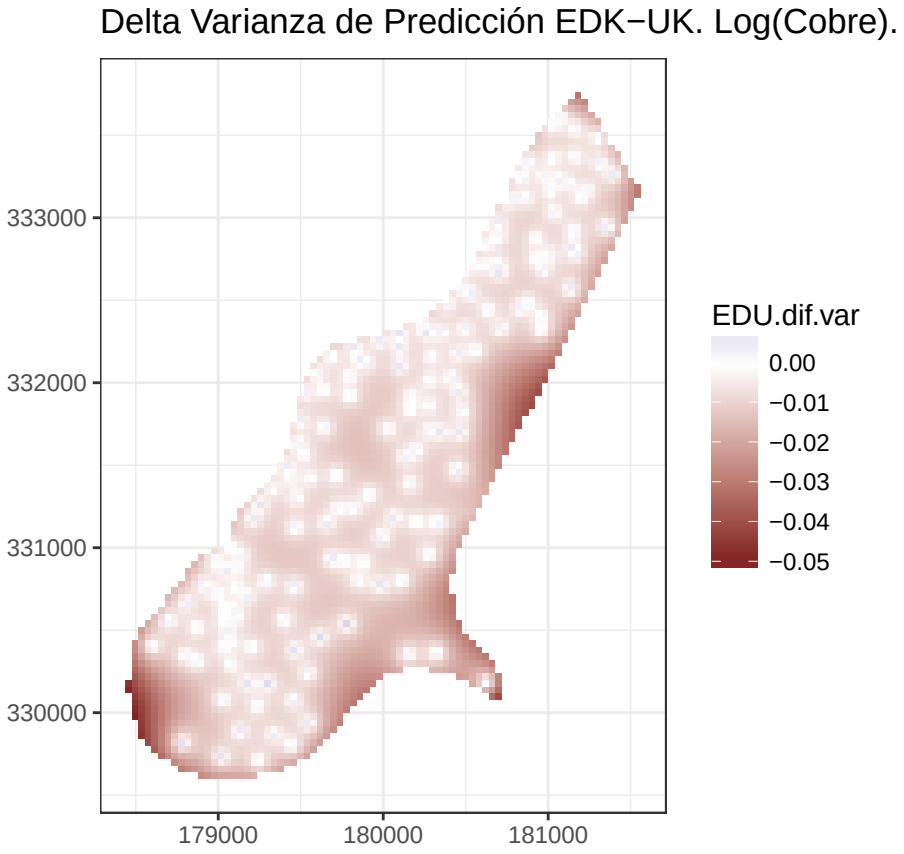
```
##        x                y           var1.pred       var1.var
##  Min.   :178460   Min.   :329620   Min.   :2.775   Min.   :0.08361
##  1st Qu.:179420   1st Qu.:330460   1st Qu.:3.191   1st Qu.:0.10879
##  Median :179980   Median :331220   Median :3.367   Median :0.12083
##  Mean   :179985   Mean   :331348   Mean   :3.447   Mean   :0.13192
##  3rd Qu.:180580   3rd Qu.:332140   3rd Qu.:3.637   3rd Qu.:0.14536
##  Max.   :181540   Max.   :333740   Max.   :4.625   Max.   :0.26575
##      U.pred          U.var          UO.dif.pred          UO.dif.var
##  Min.   :2.772   Min.   :0.07894   Min.   :-0.444217   Min.   :-0.061627
##  1st Qu.:3.191   1st Qu.:0.10212   1st Qu.:-0.034734   1st Qu.:-0.013732
##  Median :3.335   Median :0.11256   Median : 0.003102   Median :-0.008372
##  Mean   :3.424   Mean   :0.11955   Mean   :-0.023218   Mean   :-0.012369
```

```
##  3rd Qu.:3.610   3rd Qu.:0.13166   3rd Qu.: 0.019132   3rd Qu.:-0.006629
##  Max.   :4.614   Max.   :0.20703   Max.   : 0.193378   Max.   :-0.004670
##     ED.pred          ED.var         EDU.dif.pred        EDU.dif.var
##  Min.   :2.434   Min.   :0.08370   Min.   :-0.527312   Min.   :-0.050320
##  1st Qu.:3.181   1st Qu.:0.09752   1st Qu.:-0.042246   1st Qu.:-0.012627
##  Median :3.362   Median :0.10458   Median :-0.005182   Median :-0.008097
##  Mean   :3.434   Mean   :0.10975   Mean   : 0.009642   Mean   :-0.009799
##  3rd Qu.:3.699   3rd Qu.:0.11863   3rd Qu.: 0.026527   3rd Qu.:-0.004845
##  Max.   :4.545   Max.   :0.16252   Max.   : 0.757908   Max.   : 0.009885
```

```
ggplot(df.lCu.preds, aes(x,y,fill=EDU.dif.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradient2() +
  labs(x=NULL,y=NULL,
       title='Delta Predicción EDK-UK. Log(Cobre).')
```




```
ggplot(df.lCu.preds, aes(x,y,fill=EDU.dif.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradient2() +
  labs(x=NULL,y=NULL,
       title='Delta Varianza de Predicción EDK-UK. Log(Cobre).')
```



Conclusión: No existen grandes diferencias pero Kriging con Deriva Externa sobre la variable Distancia al Río es preferible a Universal Kriging por mostrar menor varianza de las predicciones. Por tanto, el orden de preferencia actual sería: EDK (Distancia al Río) > UK > OK

14 Kriging Residual Directo

14.1 Ajuste lineal y cálculo de residuos

```
lCu.lm.dist = lm(formula=log(copper) ~ dist, meuse)
summary(lCu.lm.dist)
```

```
##
## Call:
## lm(formula = log(copper) ~ dist, data = meuse)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.79260 -0.29425 -0.05632  0.28662  1.01837
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.96251    0.04864   81.47   <2e-16 ***
## dist        -1.69055    0.15662  -10.79   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3843 on 153 degrees of freedom
## Multiple R-squared:  0.4323, Adjusted R-squared:  0.4286
## F-statistic: 116.5 on 1 and 153 DF,  p-value: < 2.2e-16
```

```
res = residuals(lCu.lm.dist)
```

14.2 Inclusión de residuos en datos originales

```
meuse1 = data.frame(as.data.frame(meuse),'R0'=res)
head(meuse1)
```

```
##        x      y cadmium copper lead zinc  elev       dist   om ffreq soil
## 1 181072 333611    11.7     85  299 1022 7.909 0.00135803 13.6     1    1
## 2 181025 333558     8.6     81  277 1141 6.983 0.01222430 14.0     1    1
## 3 181165 333537     6.5     68  199  640 7.800 0.10302900 13.0     1    1
## 4 181298 333484     2.6     81  116  257 7.655 0.19009400  8.0     1    2
## 5 181307 333330     2.8     48  117  269 7.480 0.27709000  8.7     1    2
## 6 181390 333260     3.0     61  137  281 7.791 0.36406700  7.8     1    2
##   lime landuse dist.m        R0
## 1    1      Ah     50 0.4824349
## 2    1      Ah     30 0.4526028
## 3    1      Ah    150 0.4311715
## 4    0      Ga    270 0.7533010
## 5    0      Ah    380 0.3771242
## 6    0      Ga    470 0.7638363
```

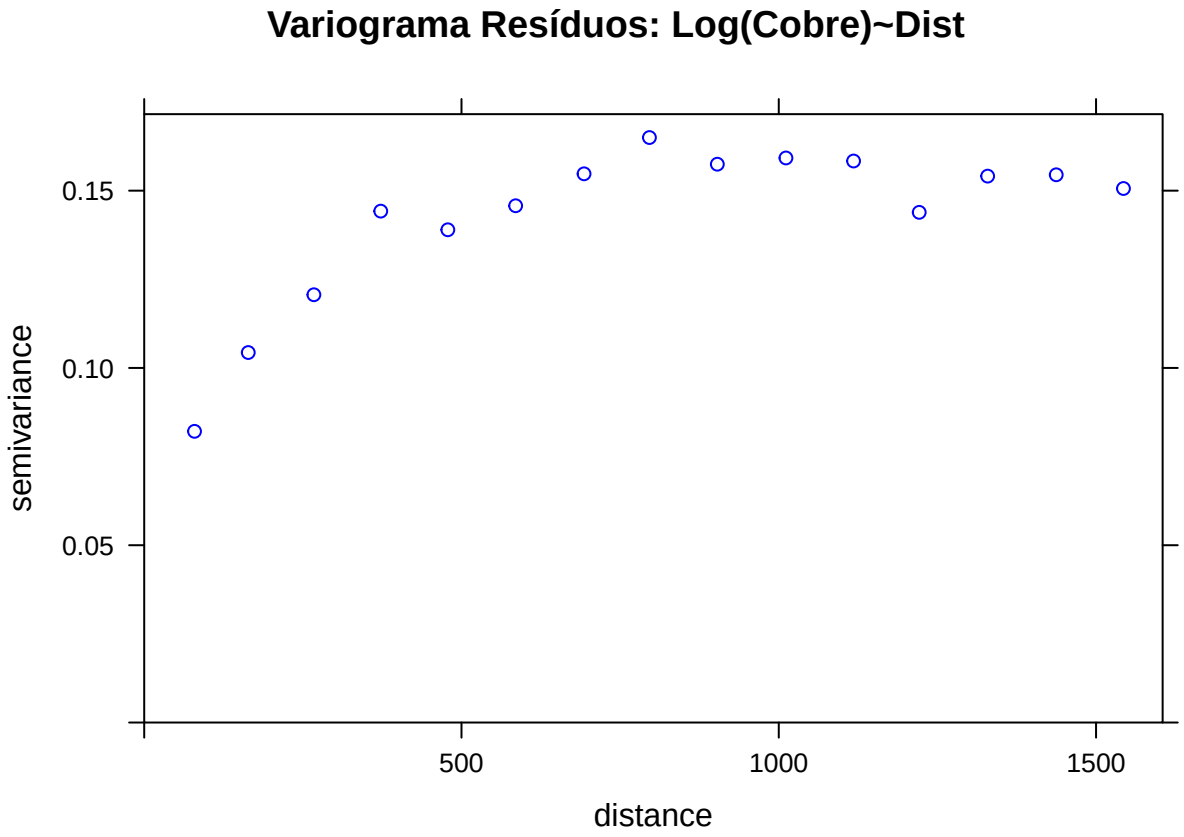
```
coordinates(meuse1) = ~x+y
```

14.3 Variograma de resíduos

```
(lCu.R0.vgm = variogram(R0~1,meuse1))
```

```
##     np       dist      gamma dir.hor dir.ver   id
## 1   57   79.29244 0.08212415       0       0 var1
## 2  299  163.97367 0.10434468       0       0 var1
## 3  419  267.36483 0.12064817       0       0 var1
## 4  457  372.73542 0.14421991       0       0 var1
## 5  547  478.47670 0.13896273       0       0 var1
## 6  533  585.34058 0.14574394       0       0 var1
## 7  574  693.14526 0.15474880       0       0 var1
## 8  564  796.18365 0.16497612       0       0 var1
## 9  589  903.14650 0.15746204       0       0 var1
## 10 543 1011.29177 0.15921954       0       0 var1
## 11 500 1117.86235 0.15836408       0       0 var1
## 12 477 1221.32810 0.14388508       0       0 var1
## 13 452 1329.16407 0.15409336       0       0 var1
## 14 457 1437.25620 0.15447101       0       0 var1
## 15 415 1543.20248 0.15061042       0       0 var1
```

```
plot(lCu.R0.vgm, col="blue",main="Variograma Resíduos: Log(Cobre)~Dist")
```



14.4 Ajuste de modelos teóricos

```
modelos = data.frame(modelos,'SSErr_R0'=NA)
for (i in c(2:15,17,18)) { # Se han excluido manualmente los que han dado error
    modelos$SSErr_R0[i] = attributes(fit.variogram(
      lCu.R0.vgm, model=vgm(0.15, modelos$short[i], 900, 0.05)))$SSErr
}
```

```
## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges =
## fit.ranges, : singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.R0.vgm, model = vgm(0.15, modelos$short[i], :
## No convergence after 200 iterations: try different initial values?
```

```
## Warning in fit.variogram(lCu.R0.vgm, model = vgm(0.15, modelos$short[i], :
## singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.R0.vgm, model = vgm(0.15, modelos$short[i], :
## singular model in variogram fit
```

```
## Warning in fit.variogram(lCu.R0.vgm, model = vgm(0.15, modelos$short[i], :
## singular model in variogram fit
```

```
modelos
```

```
##    short                                    long        SSErr
## 1    Nug                           Nug (nugget)           NA
```

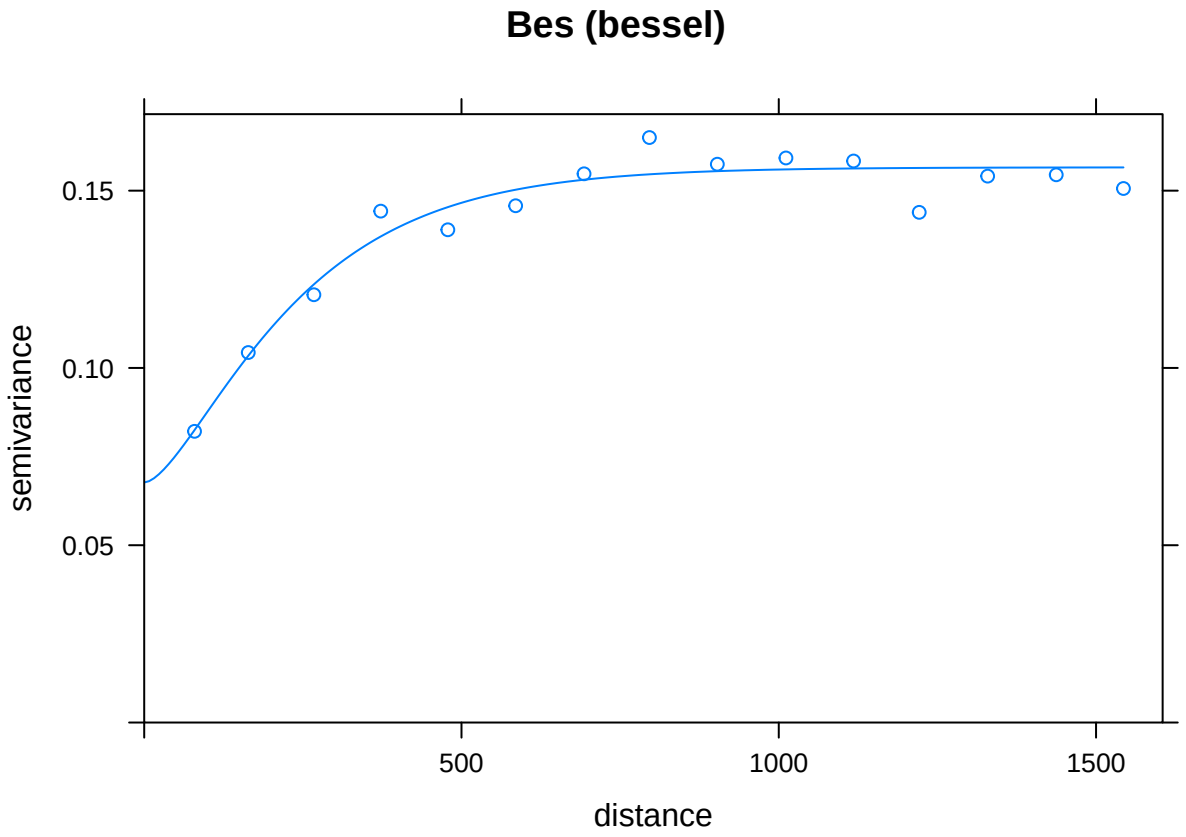
```
## 2    Exp                          Exp (exponential) 3.709119e-06
## 3    Sph                            Sph (spherical) 2.266095e-06
## 4    Gau                             Gau (gaussian) 2.951113e-06
## 5    Exc        Exclass (Exponential class/stable) 9.800221e-06
## 6    Mat                               Mat (Matern) 3.709119e-06
## 7    Ste Mat (Matern, M. Stein's parameterization) 3.709119e-06
## 8    Cir                             Cir (circular) 2.436197e-06
## 9    Lin                               Lin (linear) 3.102196e-06
## 10   Bes                               Bes (bessel) 2.645632e-06
## 11   Pen                      Pen (pentaspherical) 2.171773e-06
## 12   Per                             Per (periodic) 1.953203e-04
## 13   Wav                                 Wav (wave) 4.001240e-05
## 14   Hol                                 Hol (hole) 5.420405e-04
## 15   Log                         Log (logarithmic) 8.887894e-02
## 16   Pow                               Pow (power)           NA
## 17   Spl                             Spl (spline) 2.707089e+09
## 18   Leg                           Leg (Legendre) 1.532755e-02
## 19   Err                  Err (Measurement error)           NA
## 20   Int                            Int (Intercept)           NA
##       SSErr_res  SSErr_rdist     SSErr_R0
## 1            NA           NA           NA
## 2  1.107682e-06 5.383024e-07 5.383024e-07
## 3  1.976535e-06 7.447525e-07 7.447523e-07
## 4  2.626598e-06 8.346376e-07 8.346381e-07
## 5  1.369745e-06 8.529689e-07 8.529689e-07
## 6  1.107682e-06 5.383024e-07 5.383024e-07
## 7  1.107682e-06 5.383024e-07 5.383024e-07
## 8  2.252892e-06 8.145562e-07 8.145562e-07
## 9  3.068402e-06 1.376126e-06 1.376126e-06
## 10 1.287015e-06 5.253159e-07 5.253159e-07
## 11 1.724752e-06 6.628005e-07 6.628008e-07
## 12 6.934794e-05 2.919926e-05 2.919926e-05
## 13 1.357247e-05 2.727819e-06 9.265158e-06
## 14 8.708338e-05 3.624011e-05 3.612345e-05
## 15 6.431726e-04 1.141330e-04 1.141328e-04
## 16           NA           NA           NA
## 17 1.732537e+09 1.832742e+08 9.745523e+08
## 18 1.016827e-02 5.662254e-03 5.654429e-03
## 19           NA           NA           NA
## 20           NA           NA           NA
```

14.5 Mejor modelo

```
(mejor.modelo = modelos[which.min(modelos$SSErr_R0),])
```

```
##    short         long        SSErr    SSErr_res  SSErr_rdist     SSErr_R0
## 10   Bes Bes (bessel) 2.645632e-06 1.287015e-06 5.253159e-07 5.253159e-07
```

```
lCu.R0.fit = fit.variogram(lCu.R0.vgm,
               model = vgm(0.15, mejor.modelo$short, 900, 0.05))
plot(lCu.R0.vgm, lCu.R0.fit, main=as.character(mejor.modelo$long))
```



14.6 Predicciones

```
lCu.R0kriged = krige(R0~1,meuse1,
                     meuse.grid, model = lCu.R0.fit)
```

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

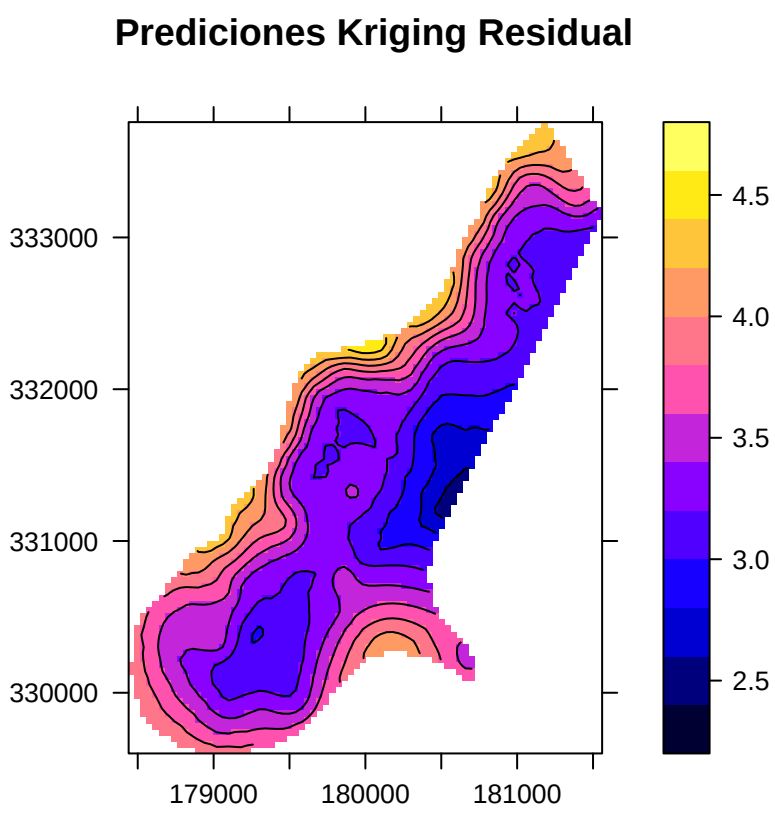
```
summary(lCu.R0kriged)
```

```
## Object of class SpatialPixelsDataFrame
## Coordinates:
##      min    max
## x 178460 181540
## y 329620 333740
## Is projected: NA
## proj4string : [NA]
## Number of points: 3103
## Grid attributes:
##   cellcentre.offset cellsize cells.dim
## x            178460       40        78
## y            329620       40       104
## Data attributes:
##    var1.pred           var1.var
##  Min.   :-0.52791   Min.   :0.08370
##  1st Qu.:-0.17737   1st Qu.:0.09747
```

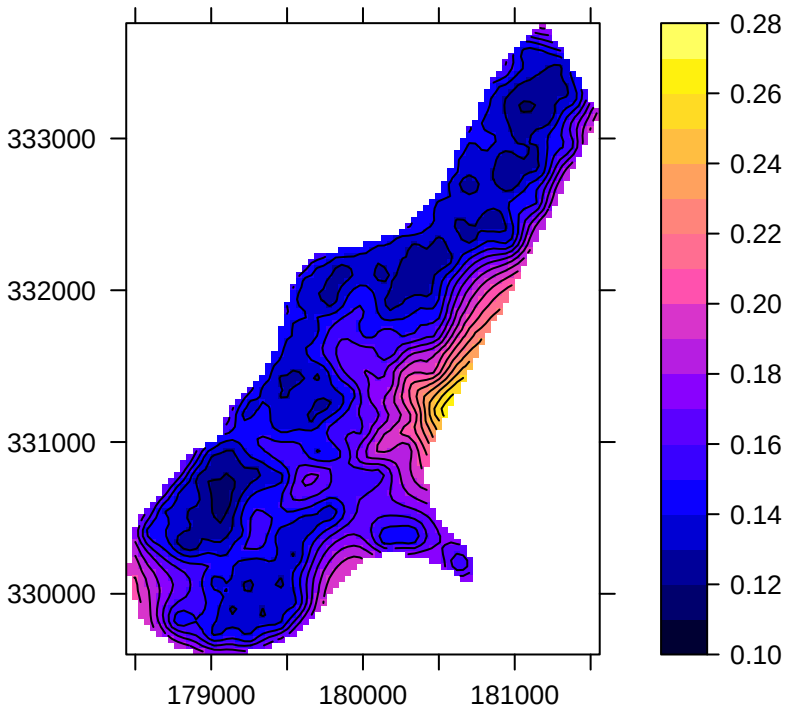
```
##  Median :-0.04672   Median :0.10448
##  Mean   :-0.02500   Mean   :0.10920
##  3rd Qu.: 0.14008   3rd Qu.:0.11812
##  Max.   : 0.57773   Max.   :0.15576
```

```
lCu.lm.dist.pred = predict(lCu.lm.dist, meuse.grid) # Predicción Deriva
lCu.R0kriged.pred = lCu.R0kriged@data$var1.pred     # Predicción Residuos
lCu.dR0.pred = lCu.lm.dist.pred + lCu.R0kriged.pred # Predicción Conjunta

lCu.lm.dist.var = predict(lCu.lm.dist, meuse.grid,se.fit = TRUE)$se.fit
lCu.R0kriged.var = lCu.R0kriged@data$var1.var
lCu.dR0.var = lCu.lm.dist.var + lCu.R0kriged.var

meuse1.grid=meuse.grid
meuse1.grid@data=cbind(meuse1.grid@data,lCu.dR0.pred,lCu.dR0.var)
```

```
spplot(meuse1.grid, zcol="lCu.dR0.pred", pretty=T, contour=T,
       col.regions=bpy.colors(64),
       main="Prediciones Kriging Residual", scales=list(draw=T))
```



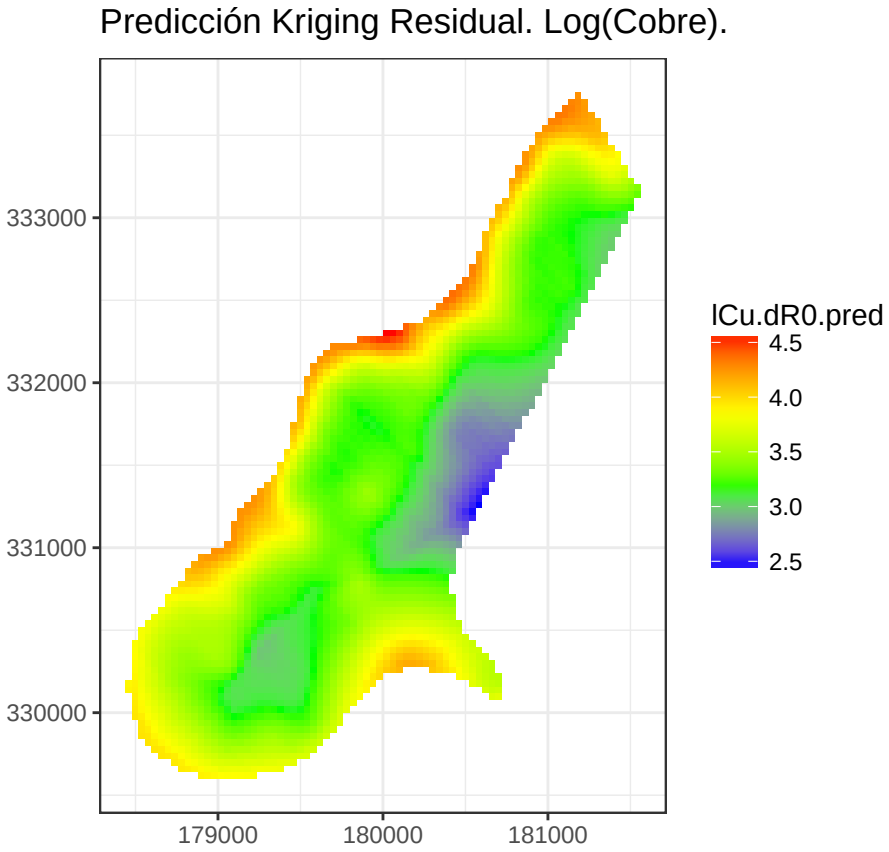

```
spplot(meuse1.grid, zcol="lCu.dR0.var", pretty=T, contour=T,
       col.regions=bpy.colors(64),
       main="Varianza Prediciones Kriging Residual", scales=list(draw=T))
```



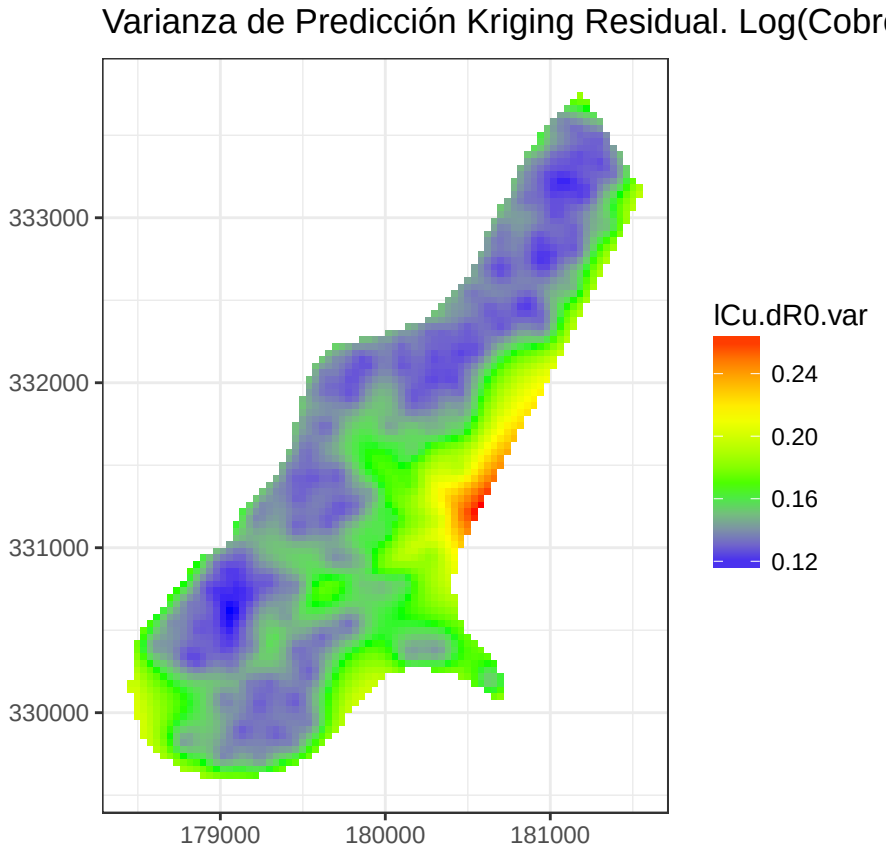

```
df.meuse1.grid = as.data.frame(meuse1.grid)
head(df.meuse1.grid)
```

```
##   part.a part.b      dist     soil       ffreq lCu.dR0.pred lCu.dR0.var
## 1      1      0 0.0000000 calcáreo cada 2 años     4.256912   0.1834901
## 2      1      0 0.0000000 calcáreo cada 2 años     4.294004   0.1713494
## 3      1      0 0.0122243 calcáreo cada 2 años     4.272068   0.1739989
## 4      1      0 0.0434678 calcáreo cada 2 años     4.215448   0.1746580
## 5      1      0 0.0000000 calcáreo cada 2 años     4.322968   0.1566332
## 6      1      0 0.0122243 calcáreo cada 2 años     4.304508   0.1597092
##        x      y
## 1 181180 333740
## 2 181140 333700
## 3 181180 333700
## 4 181220 333700
## 5 181100 333660
## 6 181140 333660
```

```
ggplot(df.meuse1.grid, aes(x,y,fill=lCu.dR0.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Predicción Kriging Residual. Log(Cobre).')
```




```
ggplot(df.meuse1.grid, aes(x,y,fill=lCu.dR0.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradientn(colors=c('blue','green','yellow','red')) +
  labs(x=NULL,y=NULL,
       title='Varianza de Predicción Kriging Residual. Log(Cobre).')
```



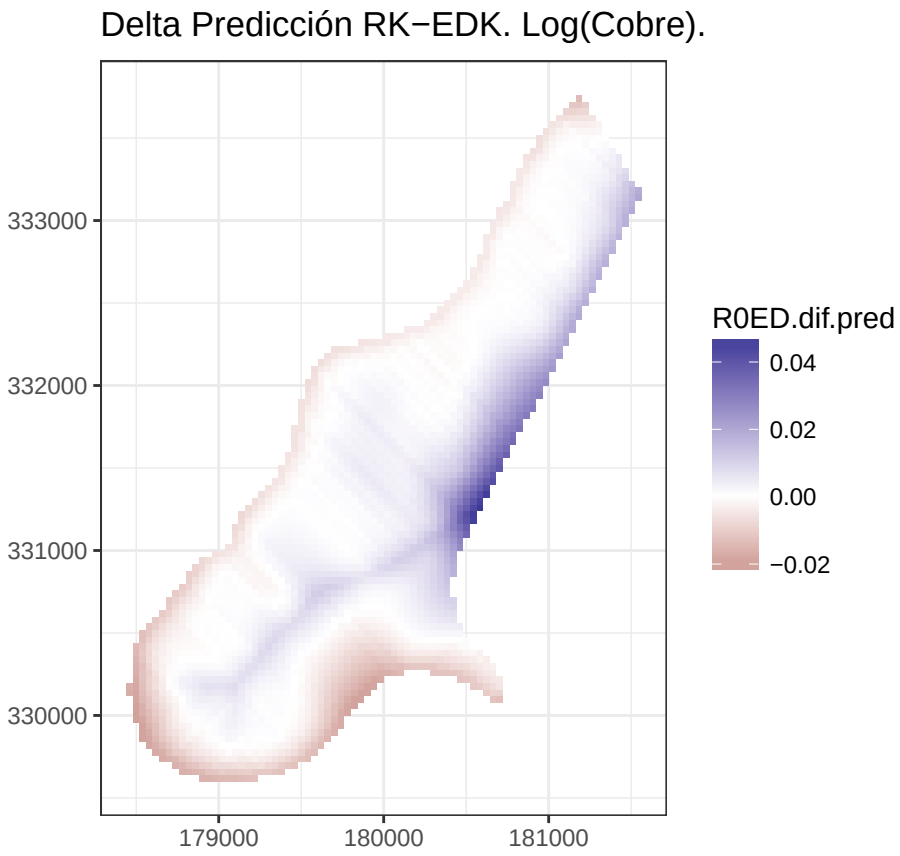
14.7 Comparativa

```
df.lCu.preds = data.frame(df.lCu.kriged,
  U.pred = lCu.Ukriged$var1.pred, U.var = lCu.Ukriged$var1.var,
  UO.dif.pred = lCu.Ukriged$var1.pred - lCu.kriged$var1.pred,
  UO.dif.var = lCu.Ukriged$var1.var - lCu.kriged$var1.var,
  ED.pred = lCu.EDkriged$var1.pred, ED.var = lCu.EDkriged$var1.var,
  EDU.dif.pred = lCu.EDkriged$var1.pred - lCu.Ukriged$var1.pred,
  EDU.dif.var = lCu.EDkriged$var1.var - lCu.Ukriged$var1.var,
  RO.pred = lCu.dRO.pred, RO.var = lCu.dRO.var,
  ROED.dif.pred = lCu.dRO.pred - lCu.EDkriged$var1.pred,
  ROED.dif.var = lCu.dRO.var - lCu.EDkriged$var1.var
  )

summary(df.lCu.preds)
```

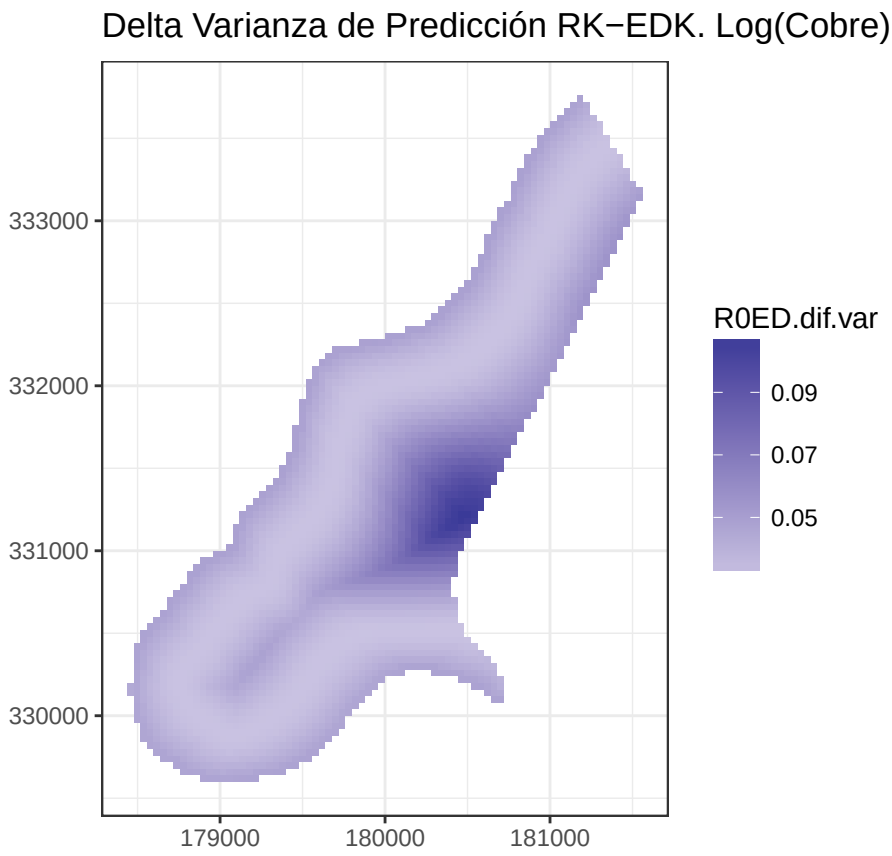
```
##        x                y            var1.pred        var1.var
##  Min.   :178460   Min.   :329620   Min.   :2.775   Min.   :0.08361
##  1st Qu.:179420   1st Qu.:330460   1st Qu.:3.191   1st Qu.:0.10879
##  Median :179980   Median :331220   Median :3.367   Median :0.12083
##  Mean   :179985   Mean   :331348   Mean   :3.447   Mean   :0.13192
##  3rd Qu.:180580   3rd Qu.:332140   3rd Qu.:3.637   3rd Qu.:0.14536
##  Max.   :181540   Max.   :333740   Max.   :4.625   Max.   :0.26575
##      U.pred           U.var          UO.dif.pred          UO.dif.var
##  Min.   :2.772   Min.   :0.07894   Min.   :-0.444217   Min.   :-0.061627
```

```
##  1st Qu.:3.191   1st Qu.:0.10212   1st Qu.:-0.034734   1st Qu.:-0.013732
##  Median :3.335   Median :0.11256   Median : 0.003102   Median :-0.008372
##  Mean   :3.424   Mean   :0.11955   Mean   :-0.023218   Mean   :-0.012369
##  3rd Qu.:3.610   3rd Qu.:0.13166   3rd Qu.: 0.019132   3rd Qu.:-0.006629
##  Max.   :4.614   Max.   :0.20703   Max.   : 0.193378   Max.   :-0.004670
##     ED.pred          ED.var          EDU.dif.pred        EDU.dif.var
##  Min.   :2.434   Min.   :0.08370   Min.   :-0.527312   Min.   :-0.050320
##  1st Qu.:3.181   1st Qu.:0.09752   1st Qu.:-0.042246   1st Qu.:-0.012627
##  Median :3.362   Median :0.10458   Median :-0.005182   Median :-0.008097
##  Mean   :3.434   Mean   :0.10975   Mean   : 0.009642   Mean   :-0.009799
##  3rd Qu.:3.699   3rd Qu.:0.11863   3rd Qu.: 0.026527   3rd Qu.:-0.004845
##  Max.   :4.545   Max.   :0.16252   Max.   : 0.757908   Max.   : 0.009885
##     R0.pred          R0.var        R0ED.dif.pred         R0ED.dif.var
##  Min.   :2.482   Min.   :0.1151   Min.   :-0.0202130   Min.   :0.03072
##  1st Qu.:3.186   1st Qu.:0.1350   1st Qu.:-0.0017122   1st Qu.:0.03314
##  Median :3.364   Median :0.1467   Median : 0.0007157   Median :0.03939
##  Mean   :3.435   Mean   :0.1533   Mean   : 0.0016831   Mean   :0.04357
##  3rd Qu.:3.695   3rd Qu.:0.1654   3rd Qu.: 0.0038744   3rd Qu.:0.04709
##  Max.   :4.540   Max.   :0.2645   Max.   : 0.0480805   Max.   :0.10804
```

```
ggplot(df.lCu.preds, aes(x,y,fill=R0ED.dif.pred)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradient2() +
  labs(x=NULL,y=NULL,
       title='Delta Predicción RK-EDK. Log(Cobre).')
```



```
ggplot(df.lCu.preds, aes(x,y,fill=R0ED.dif.var)) +
  geom_raster() + coord_equal() + theme_bw() +
  scale_fill_gradient2() +
  labs(x=NULL,y=NULL,
       title='Delta Varianza de Predicción RK-EDK. Log(Cobre).')
```



Conclusión: No existen grandes diferencias pero Kriging con Deriva Externa sobre la variable Distancia al Río es preferible al Kriging Residual Directo sobre dicha misma variable mostrando una menor varianza de las predicciones.

El orden de preferencia para los distintos métodos estudiados sería el siguiente:

EDK (Distancia al Río) > UK > OK > RK (Distancia al Río)