DSA 8020 R Session 14: Spatial Interpolation II

Whitney Huang

Contents

Rainfall Data from Parana State, Brazil

In this R lab, we analyze dry season (May-June) rainfall data collected from 143 weather stations in Parana State, Brazil, to illustrate spatial interpolation.

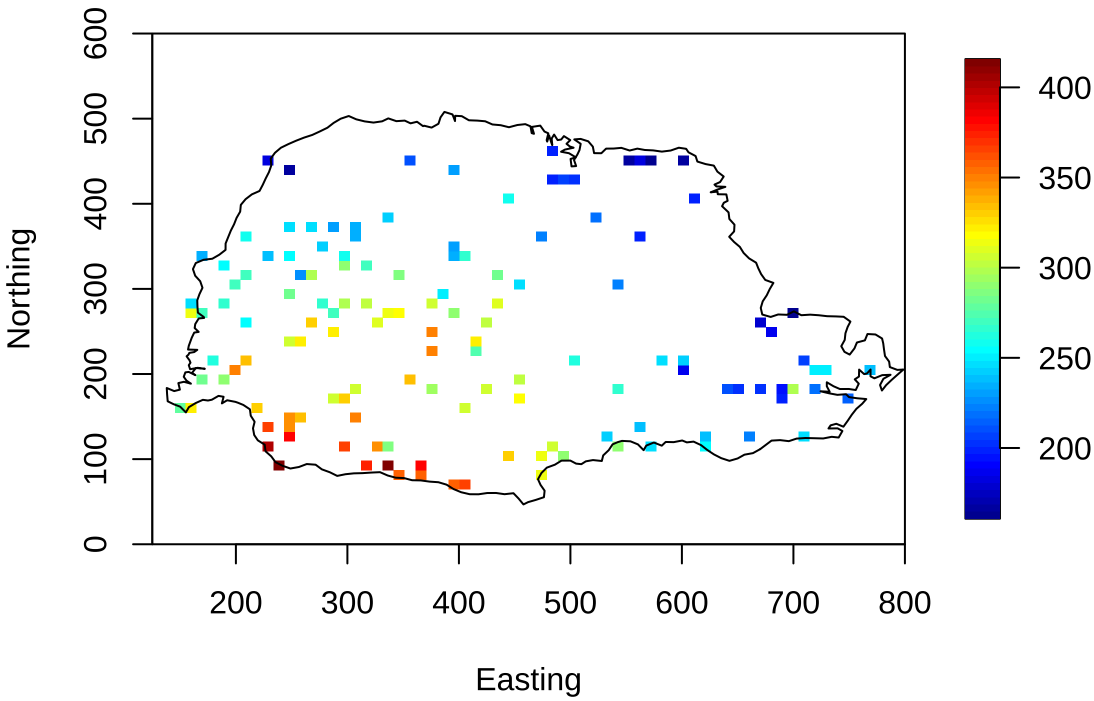
Loading and summarizing the data

```r
library(geoR)
data(parana)
summary(parana)
```

```
## Number of data points: 143
##
## Coordinates summary
##         east    north
## min 150.1220  70.3600
## max 768.5087 461.9681
##
## Distance summary
##      min      max
##   1.0000 619.4925
##
## Borders summary
##         east    north
## min 137.9873  46.7695
## max 798.6256 507.9295
##
```

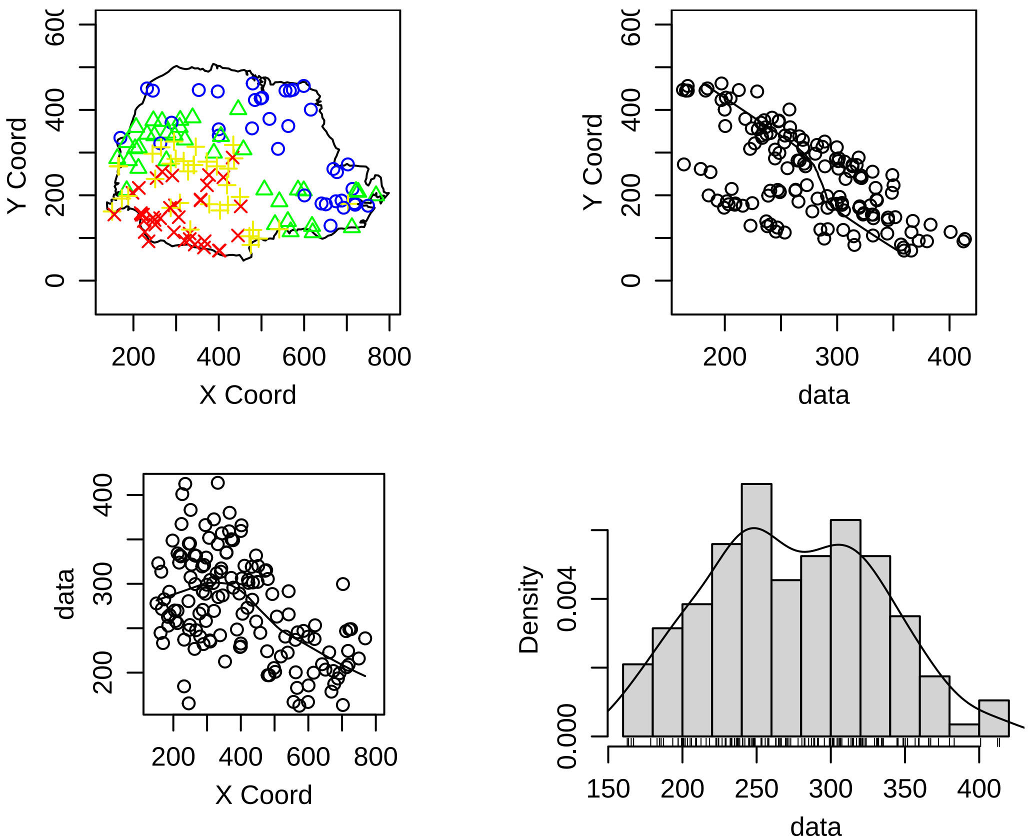
```
## Data summary
##      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 162.7700 234.1900 269.9200 274.4106 318.2300 413.7000
##
## Other elements in the geodata object
## [1] "loci.paper"
```

```r
library(fields)
quilt.plot(parana$coords, parana$data, ny = 36, ylim = c(0, 600),
           xlim = c(125, 800), xlab = "Easting", ylab = "Northing")
lines(parana$borders)
```

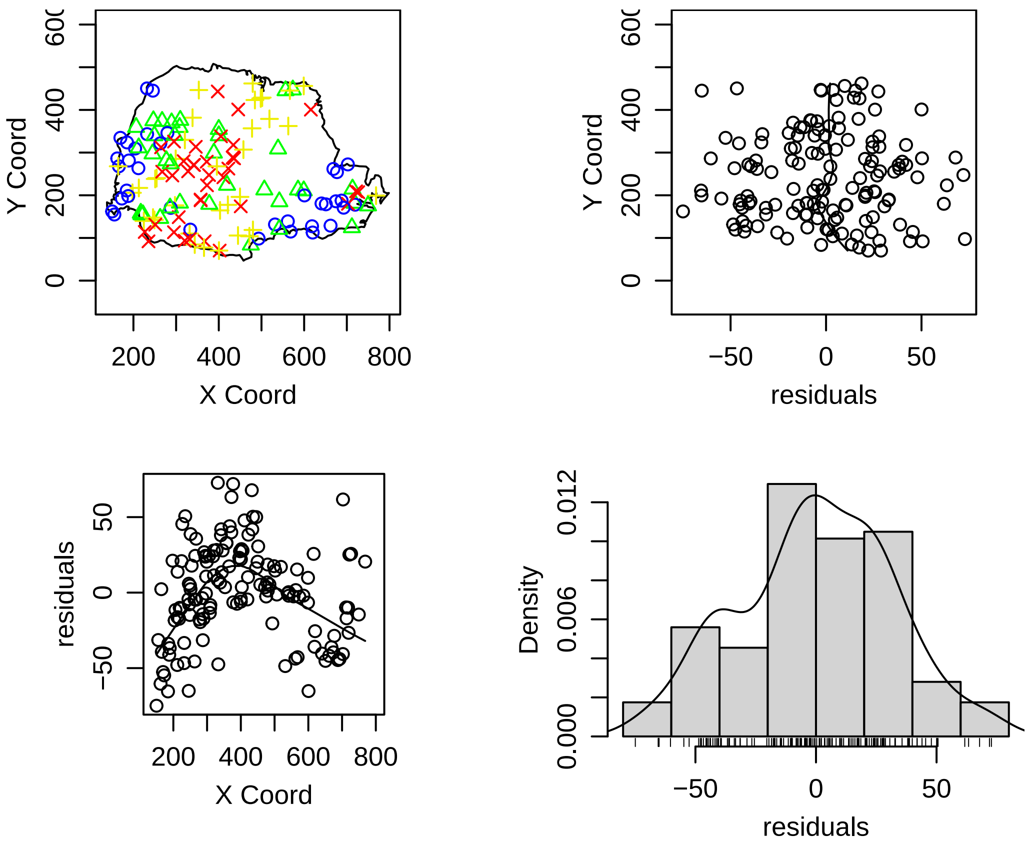


Below, we are going to examine the potential spatial trend that characterizes the large-scale spatial variation. The non-parametric LOWESS fits on the x-axis (longitude) and y-axis (latitude) suggest approximately linear trends for both longitude and latitude, which is a reasonable choice.

```r
plot(parana, lowess = TRUE)
```




```
plot(parana, trend = "1st", lowess = TRUE)
```



Variogram Analysis

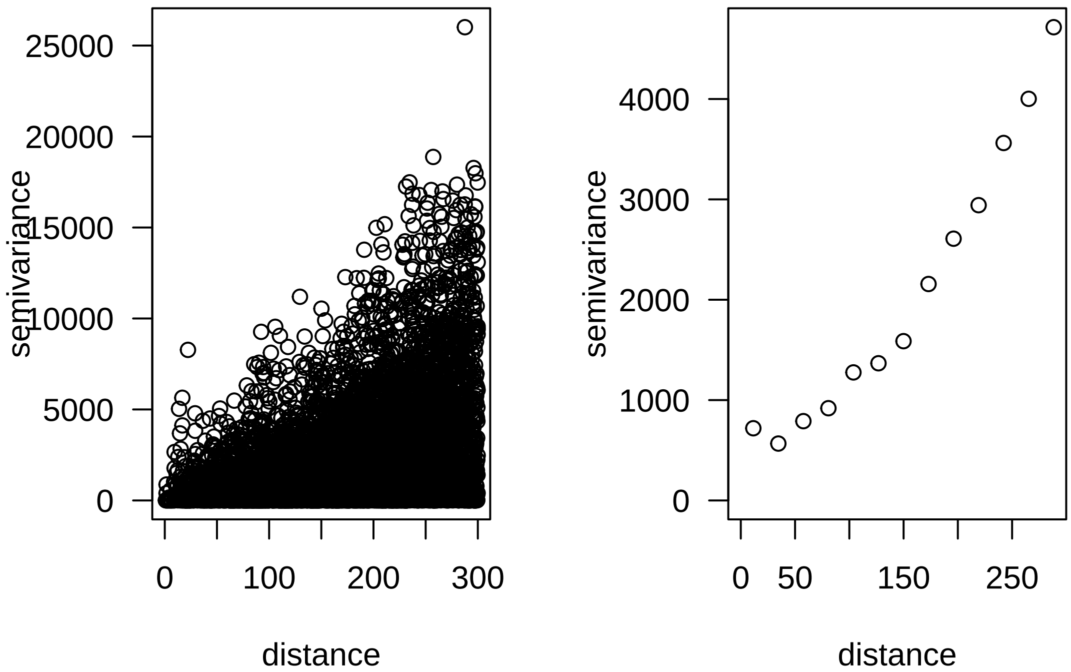
```
par(mfrow = c(1, 2), las = 1, mgp = c(3, 1, 0))
parana.vario <- variog(parana, max.dist = 300, cex = 0.5, option = "cloud")
```

```
## variog: computing omnidirectional variogram
```

```
plot(parana.vario)
parana.vario <- variog(parana, max.dist = 300, cex = 0.5)
```

```
## variog: computing omnidirectional variogram
```

```
plot(parana.vario)
```



Below, you will see the variogram with (right) and without (left) the removal of the spatial trend component.

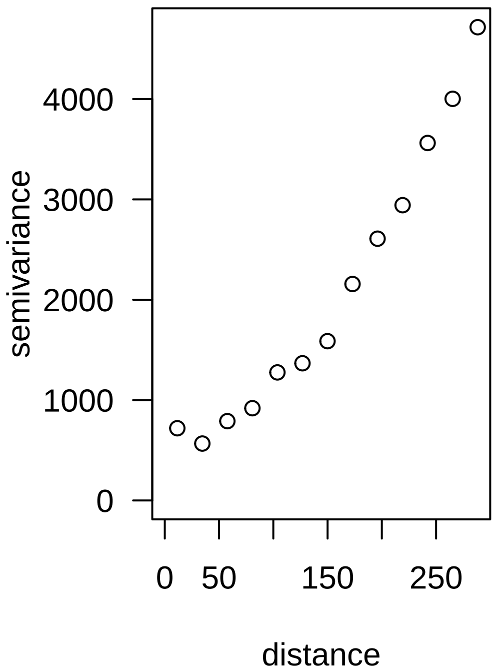
```
par(mfrow = c(1, 2), las = 1, mgp = c(3, 1, 0))
parana.vario <- variog(parana, max.dist = 300, cex = 0.5)
```

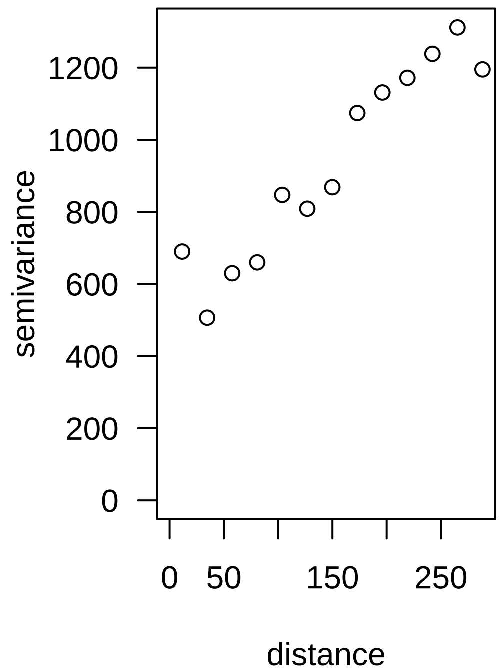
```
## variog: computing omnidirectional variogram
```

```
plot(parana.vario)
parana.variot <- variog(parana, trend = "1st",  max.dist = 300)
```

```
## variog: computing omnidirectional variogram
```

```
plot(parana.variot)
```





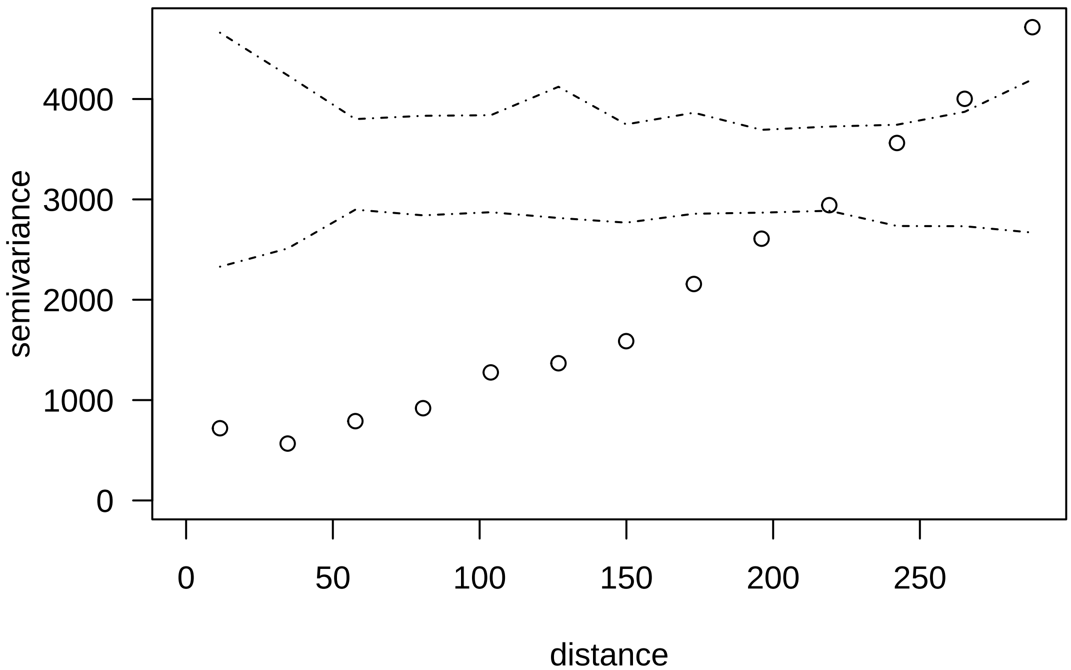

```r
par(las = 1)
parana.v <- variog(parana, max.dist = 300)
```

```
## variog: computing omnidirectional variogram
```

```r
parana.v.env <- variog.mc.env(parana, obj.variog = parana.v)
```

```
## variog.env: generating 99 simulations by permutating data values
## variog.env: computing the empirical variogram for the 99 simulations
## variog.env: computing the envelops
```

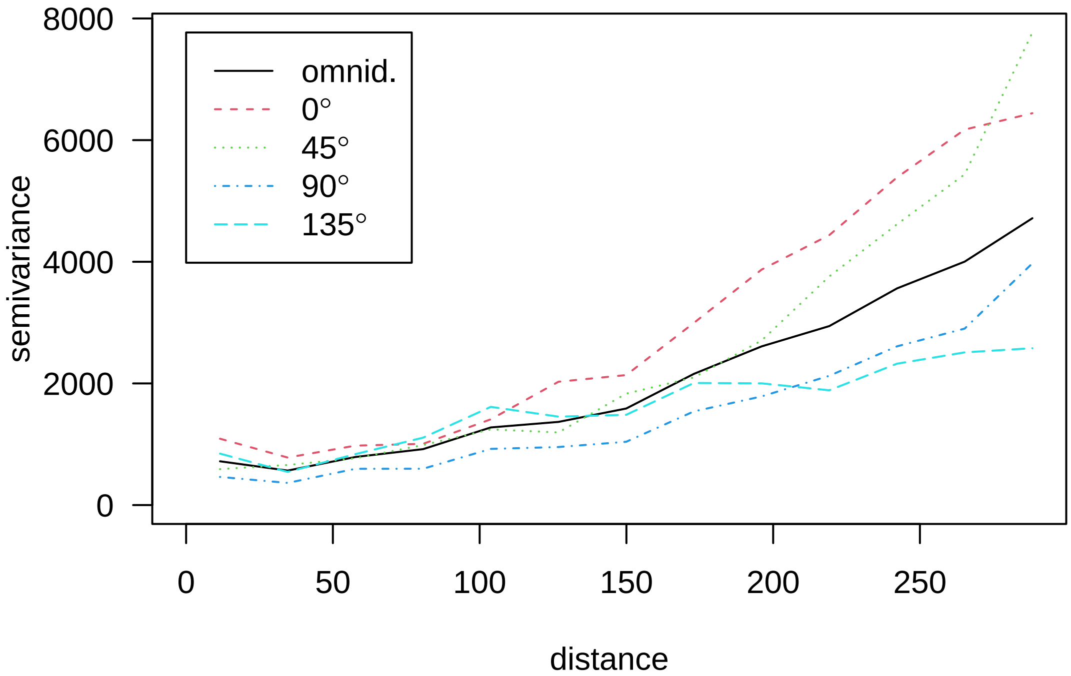
```r
plot(parana.v, env = parana.v.env)
```




```r
par(las = 1)
parana.v4 <- variog4(parana, max.dist = 300)
```

```
## variog: computing variogram for direction = 0 degrees (0 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing variogram for direction = 45 degrees (0.785 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing variogram for direction = 90 degrees (1.571 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing variogram for direction = 135 degrees (2.356 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing omnidirectional variogram
```

```r
plot(parana.v4, env = parana.v.env, omni = TRUE)
```



Parameter Estimation

```
# with linear trend
parana.vtfit.exp <- variofit(parana.variot)
```

```
## variofit: covariance model used is matern
## variofit: weights used: npairs
## variofit: minimisation function used: optim

## Warning in variofit(parana.variot): initial values not provided - running the
## default search

## variofit: searching for best initial value ... selected values:
##               sigmasq   phi      tausq    kappa
## initial.value "1311.47" "230.66" "327.87" "0.5"
## status        "est"     "est"    "est"    "fix"
## loss value: 33524269.3444707
```

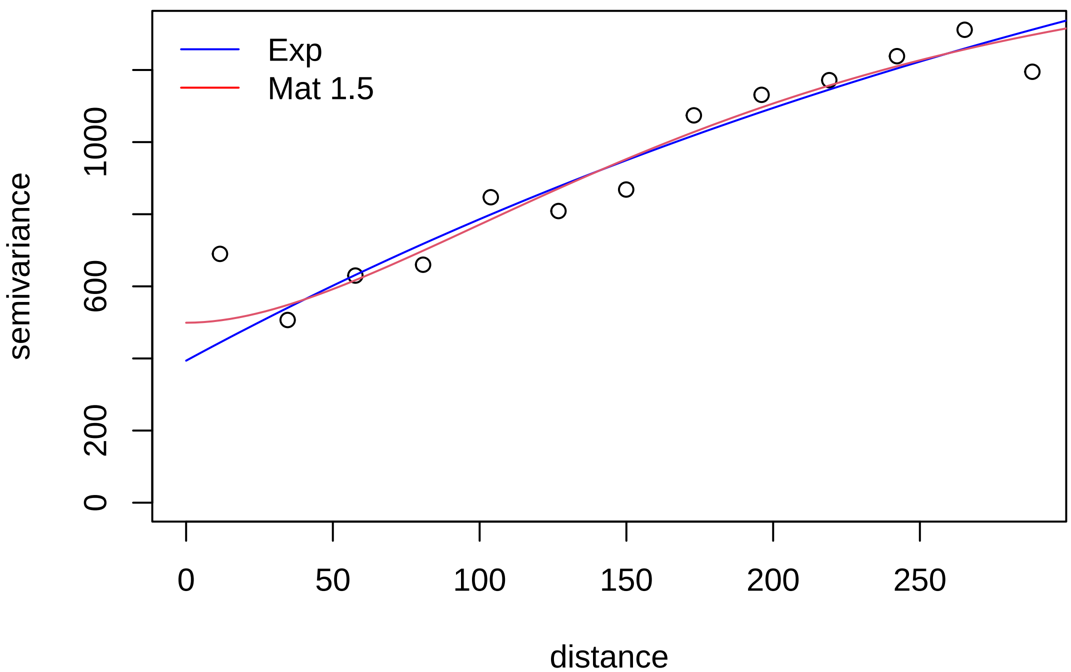
```
parana.vtfit.mat1.5  <- variofit(parana.variot, kappa = 1.5)
```

```
## variofit: covariance model used is matern
## variofit: weights used: npairs
## variofit: minimisation function used: optim

## Warning in variofit(parana.variot, kappa = 1.5): initial values not provided -
## running the default search
```

```
## variofit: searching for best initial value ... selected values:
##                 sigmasq phi      tausq    kappa
## initial.value "983.6" "138.39" "655.73" "1.5"
## status        "est"   "est"    "est"    "fix"
## loss value: 43717205.8946468
```

```r
plot(parana.variot)
lines(parana.vtfit.exp, col = "blue"); lines(parana.vtfit.mat1.5, col = 2)
legend("topleft", legend = c("Exp", "Mat 1.5"), col = c("blue", "red"),
       lty = 1, bty = "n")
```



MLE

```r
(parana.ml1 <- likfit(parana, trend = "1st", ini = c(1000, 50), nug = 100))
```

```
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
## likfit: estimated model parameters:
##      beta0      beta1      beta2      tausq    sigmasq        phi
## "416.4984" " -0.1375" " -0.3997" "385.5180" "785.6904" "184.3863"
```

```
## Practical Range with cor=0.05 for asymptotic range: 552.3719
##
## likfit: maximised log-likelihood = -663.9
```

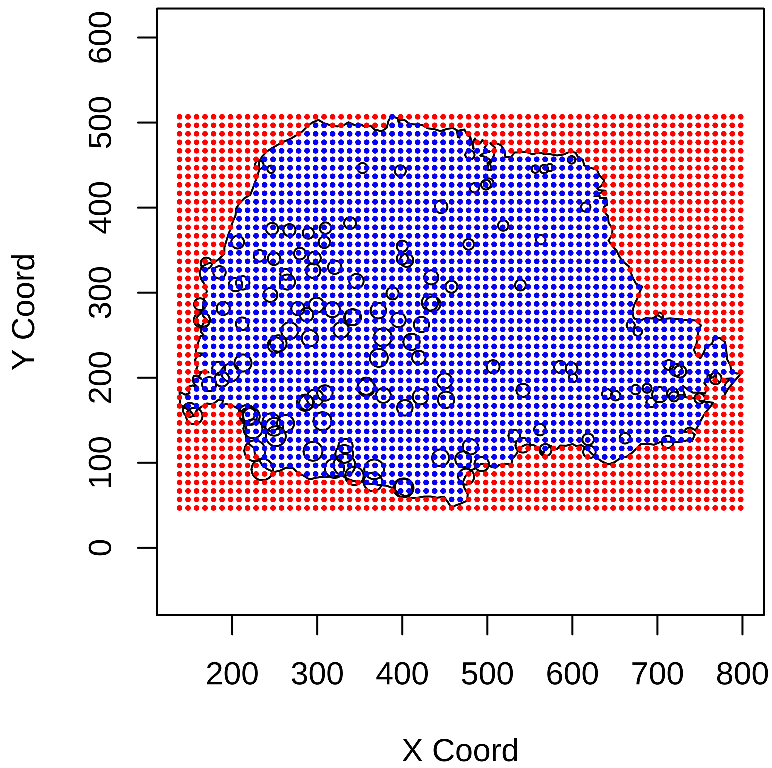
```r
(parana.ml2 <- likfit(parana, trend = "2nd", ini = c(1000, 50), nug = 100))
```

```
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
## likfit: estimated model parameters:
##      beta0      beta1      beta2      beta3      beta4      beta5      tausq
## "423.9282" "  0.0620" " -0.6360" " -0.0004" "  0.0000" "  0.0006" "381.2267"
##    sigmasq        phi
## "372.5993" " 77.5441"
## Practical Range with cor=0.05 for asymptotic range: 232.3013
##
## likfit: maximised log-likelihood = -660.2
```

Spatial Prediction

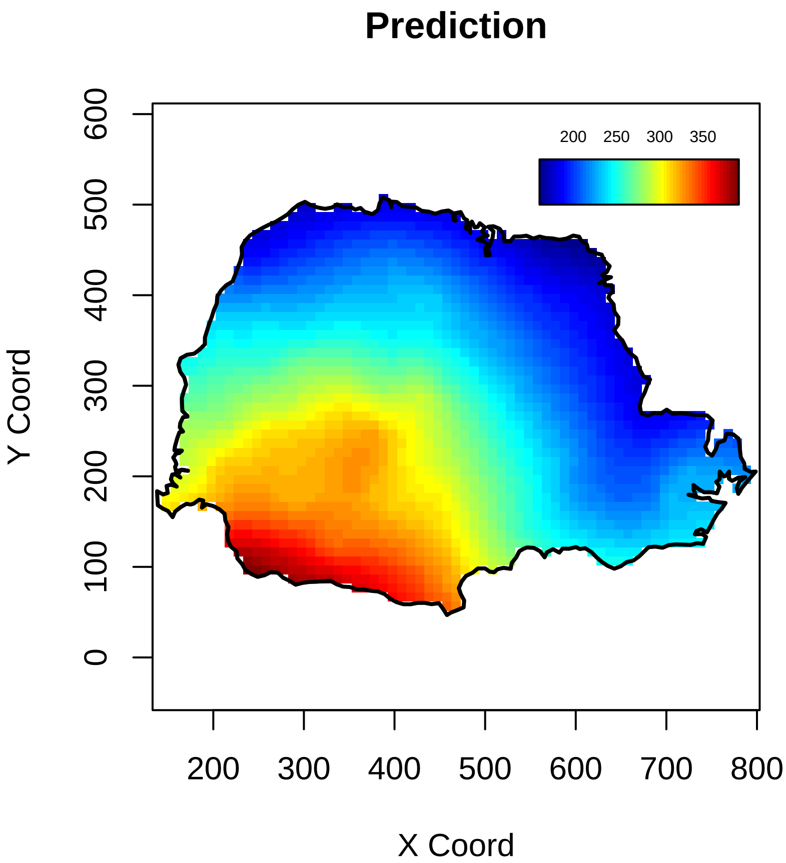
```r
parana.gr <- pred_grid(parana$borders, by = 10)
points(parana)
points(parana.gr, pch = 19, col = "red", cex = 0.25)
parana.gr0 <- locations.inside(parana.gr, parana$borders)
points(parana.gr0, pch = 19, col = "blue", cex = 0.25)
```



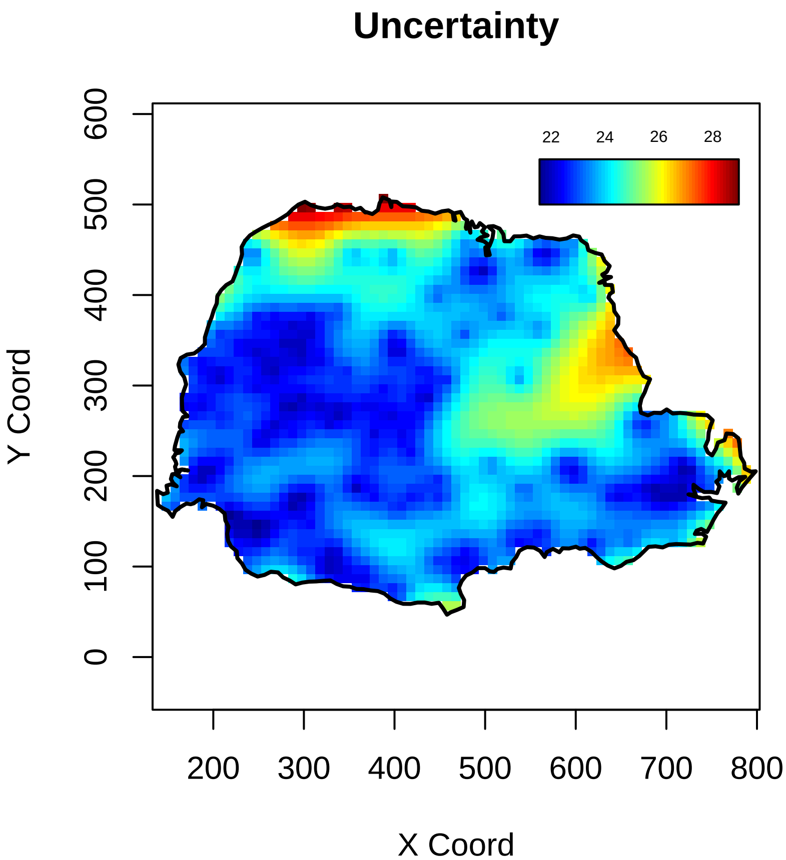

```r
KC <- krige.control(obj.m = parana.ml1, trend.d = "1st", trend.l = "1st")
OC <- output.control(simulations = TRUE, n.pred = 1000)
parana.kc <- krige.conv(parana, loc = parana.gr, krige = KC, output = OC)
```

```
## krige.conv: results will be returned only for prediction locations inside the borders
## krige.conv: model with mean given by a 1st order polynomial on the coordinates
## krige.conv: sampling from the predictive distribution (conditional simulations)
## krige.conv: Kriging performed using global neighbourhood
```

```r
## Spatial prediction and prediction uncertainty
image(parana.kc, col = tim.colors(), x.leg = c(560, 780),
      y.leg = c(500, 550), cex = 0.5, main = "Prediction")
```

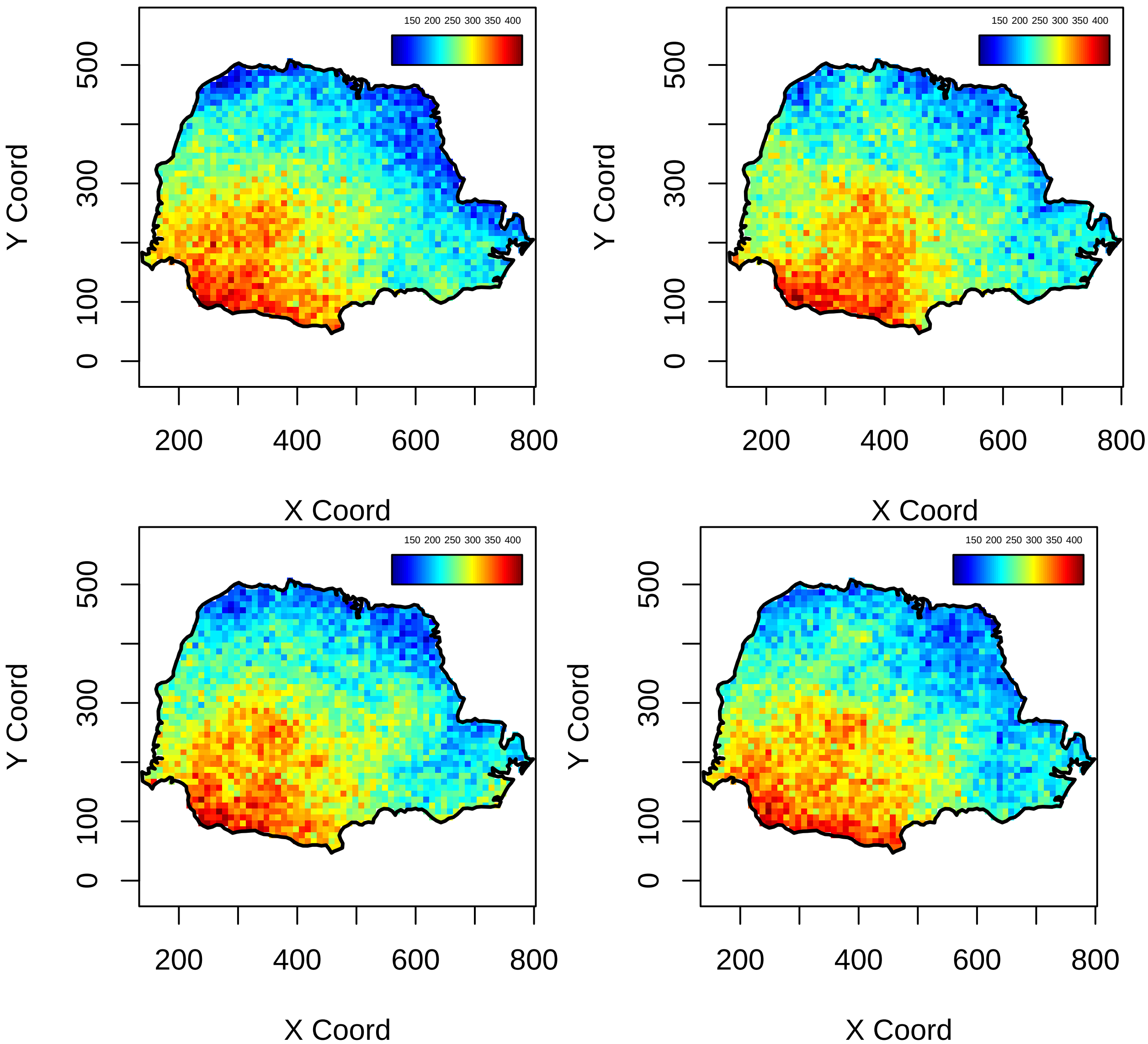



```
image(parana.kc, val = sqrt(parana.kc$krige.var), col = tim.colors(),
      x.leg = c(560, 780), y.leg = c(500, 550), cex = 0.5, main = "Uncertainty")
```



Conditional Simulations

```
for (i in 1:4){
  image(parana.kc, val = parana.kc$simulation[, i], col = tim.colors(),
      x.leg = c(560, 780), y.leg = c(500, 550), cex = 0.35,
      main = "", zlim = range(parana.kc$simulation[, 1:4]))
}
```


150 200 250 300 350 400
Y Coord
0
100
300
500
200
400
600
800
X Coord