Analysis of rainfall in the state of Paraná, Brazil

Alexandra M. Schmidt

May 2018

This data-set was used by Diggle and Ribeiro (2001) to illustrate the methods discussed in the paper. The data reported analysis was carried out using the package geoR.

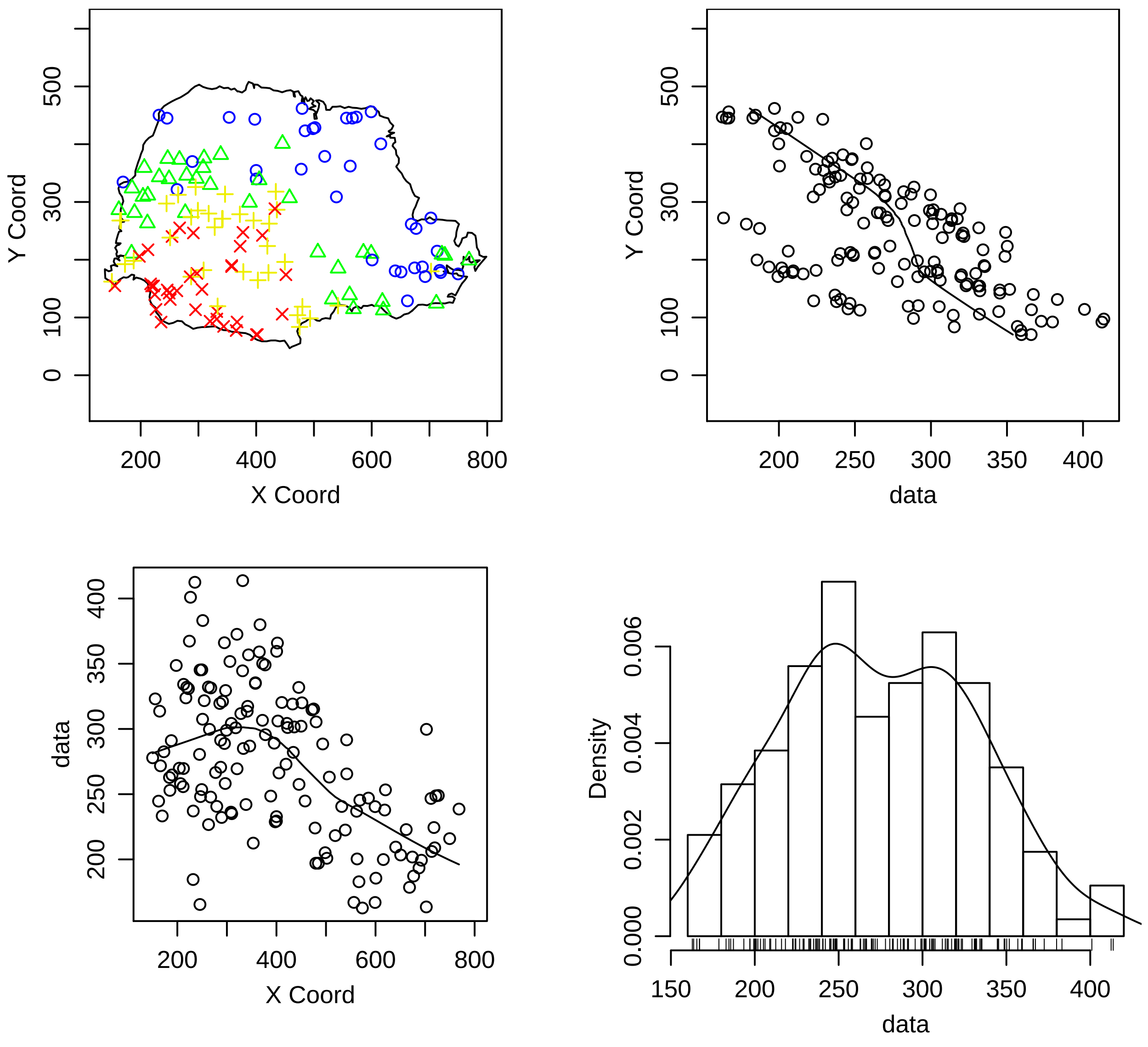
The data refers to average rainfall over different years for the period May-June (dry-season). It was collected at 143 recording stations throughout the state of Paraná, Brazil. And this analysis is based on the lecture that Paulo Justiniano Ribeiro Jr gave in the Pan-American Pan-American Advanced Study Institute on Spatio-Temporal Statistics in Búzios, Brazil in 2014.

1 Exloporatory data analysis

```
library(geoR)

## --------------------------------------------------------------
##  Analysis of Geostatistical Data
##  For an Introduction to geoR go to http://www.leg.ufpr.br/geoR
##  geoR version 1.7-5.2 (built on 2016-05-02) is now loaded
## --------------------------------------------------------------

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
```

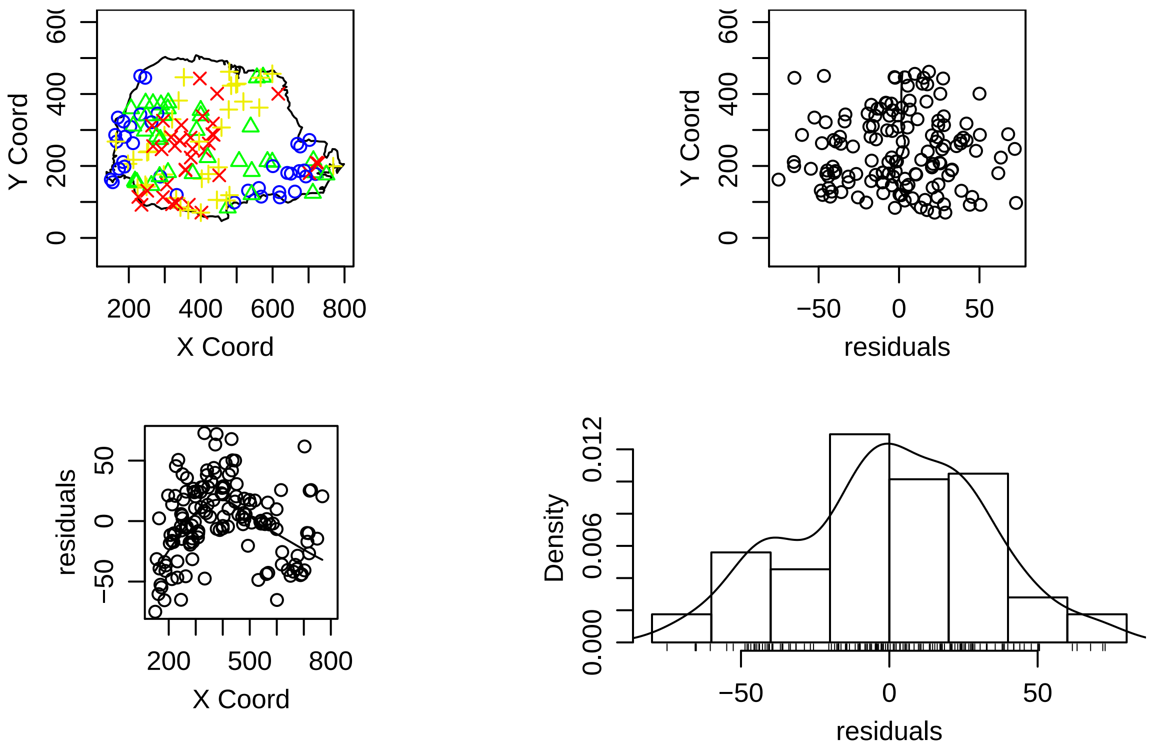
```
##
## Borders summary
##          east    north
## min 137.9873  46.7695
## max 798.6256 507.9295
##
## Data summary
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 162.7700 234.1900 269.9200 274.4106 318.2300 413.7000
##
## Other elements in the geodata object
## [1] "loci.paper"
```

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



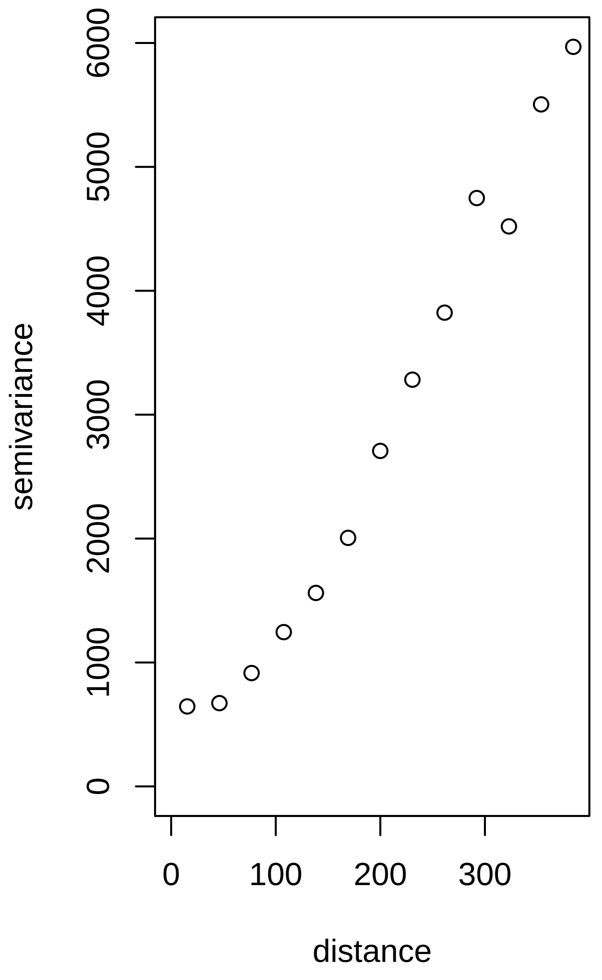
If the argument trend is provided a (ordinary) regression is fitted and the residuals are plotted. The options trend=1st defines a linear trend on the coordinates and is equivalent to trend= coords. More generally a formula on a set of covariates can be used.

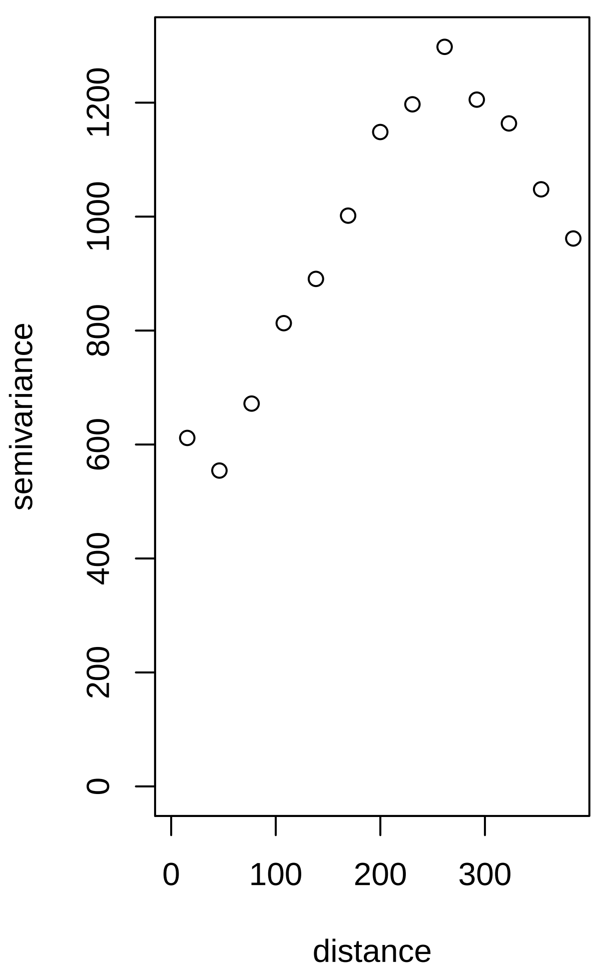
```
plot(parana, trend = "1st", lowess=TRUE)
```



The effect on the trend in the variograms are very much noticeable in this example.

```
par(mfrow=c(1,2))
parana.vario <- variog(parana, max.dist=400)

## variog: computing omnidirectional variogram

plot(parana.vario)
parana.variot <- variog(parana, trend="1st",  max.dist=400)

## variog: computing omnidirectional variogram

plot(parana.variot)
```





```
parana.vfit.exp <- variofit(parana.vario)

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

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

## variofit: searching for best initial value ... selected values:
##               sigmasq   phi      tausq kappa
## initial.value "5969.14" "246.07" "0"   "0.5"
## status        "est"     "est"    "est" "fix"
## loss value: 4612934365.43219

parana.vfit.mat1.5  <- variofit(parana.vario, kappa=1.5)
```

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

## Warning in variofit(parana.vario, 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 "5969.14" "123.04" "0"   "1.5"
## status        "est"     "est"    "est" "fix"
## loss value: 1635452671.02251

parana.vfit.sph  <- variofit(parana.vario, cov.model="sph")

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

## Warning in variofit(parana.vario, cov.model = "sph"):  initial values not provided - running
the default search

## variofit: searching for best initial value ... selected values:
##               sigmasq   phi      tausq kappa
## initial.value "4476.85" "307.59" "0"   "0.5"
## status        "est"     "est"    "est" "fix"
## loss value: 7540185095.69744

#
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 "973.53" "123.04" "324.51" "0.5"
## status        "est"    "est"    "est"    "fix"
## loss value: 109923374.036732

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 "973.53" "61.52" "324.51" "1.5"
## status        "est"    "est"   "est"    "fix"
## loss value: 115232526.189321
```

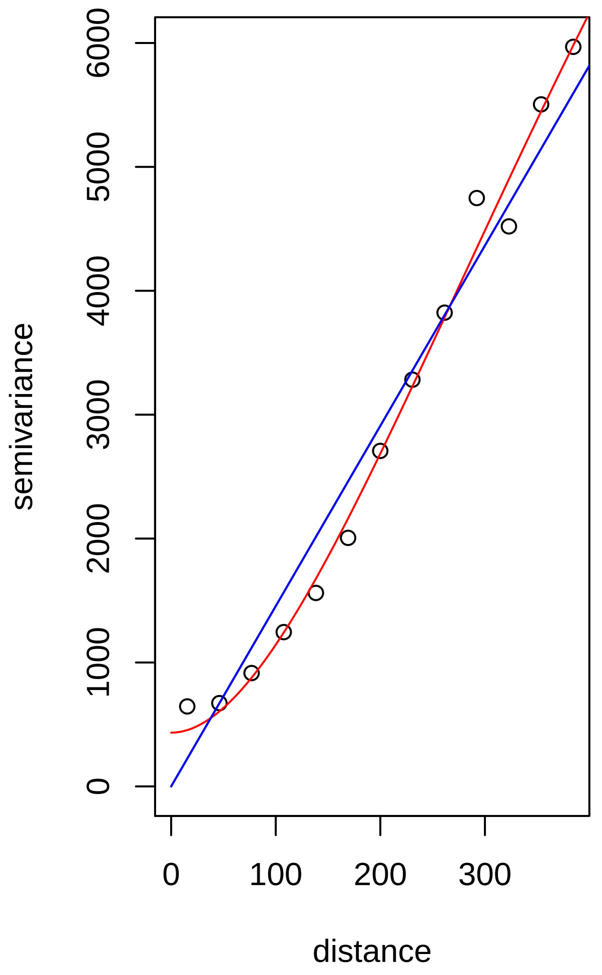
```
parana.vtfit.sph  <- variofit(parana.variot, cov.model="sph")
```

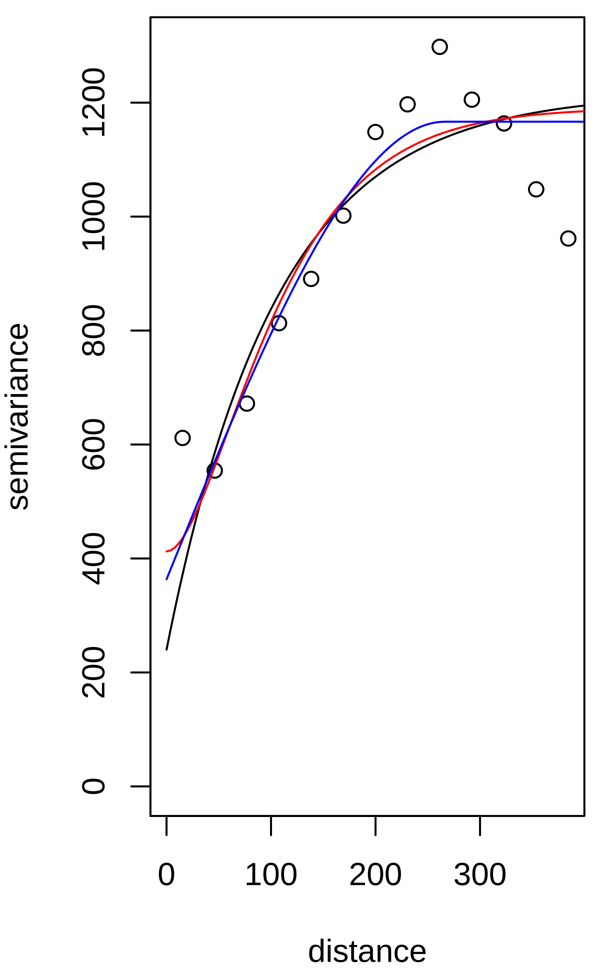
```
## variofit: covariance model used is spherical
## variofit: weights used: npairs
## variofit: minimisation function used: optim
```

```
## Warning in variofit(parana.variot, cov.model = "sph"):  initial values not provided - running
the default search
```

```
## variofit: searching for best initial value ... selected values:
##               sigmasq  phi      tausq   kappa
## initial.value "973.53" "184.55" "129.8" "0.5"
## status        "est"    "est"    "est"   "fix"
## loss value: 112750740.264926
```

```
par(mfrow=c(1,2))
plot(parana.vario)
lines(parana.vfit.exp);lines(parana.vfit.mat1.5, col=2);lines(parana.vfit.sph, col=4)
plot(parana.variot)
lines(parana.vtfit.exp);lines(parana.vtfit.mat1.5, col=2);lines(parana.vtfit.sph, col=4)
```





2 Inference

Maximum likelihood estimates are obtained fitting the model to the data (and not to the empirical variogram). Models can be compared by fitting measures.

```
(parana.ml0 <- likfit(parana, ini=c(4500,50), nug=500))

## ---------------------------------------------------------------
## 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:
##     beta    tausq  sigmasq      phi
## " 243.4" " 358.5" "9594.1" "1658.6"
## Practical Range with cor=0.05 for asymptotic range: 4968.778
##
## likfit: maximised log-likelihood = -671.6
```

```
(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
```

```
(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
## "423.9282" "  0.0620" " -0.6360" " -0.0004" "  0.0000" "  0.0006"
##      tausq    sigmasq        phi
## "381.2267" "372.5993" " 77.5441"
## Practical Range with cor=0.05 for asymptotic range: 232.3013
##
## likfit: maximised log-likelihood = -660.2
```

```
logLik(parana.ml0)
```

```
## 'log Lik.' -671.6381 (df=4)
```

```
logLik(parana.ml1)
```

```
## 'log Lik.' -663.8597 (df=6)
```

```
logLik(parana.ml2)
```

```
## 'log Lik.' -660.1756 (df=9)
```

3 Spatial interpolation

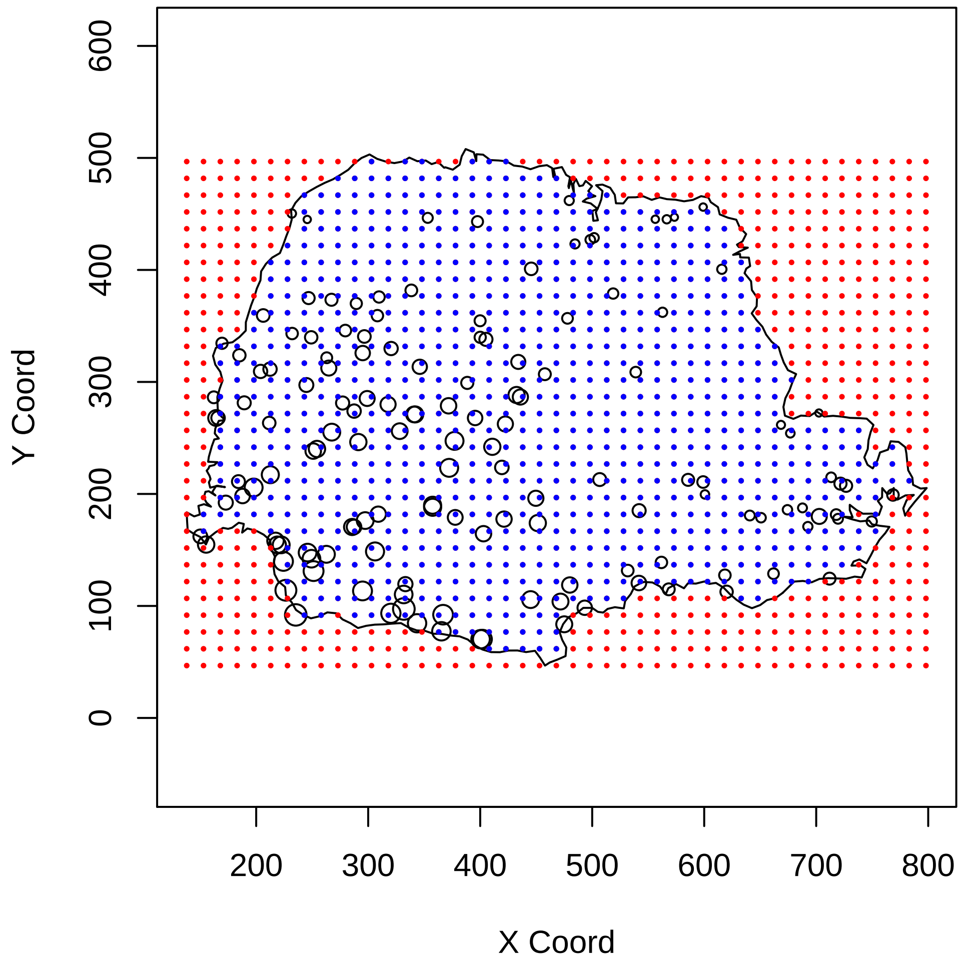
There are two functions for spatial prediction in geoR.

- `krige.conv()` performing plug-in predictions, i.e. treating estimated parameters as if they were the true values
- `krige.bayes()` for Bayesian inference and prediction

For the former, the function takes: the data, coordinates of the locations, kriging options (through krige.control()) and output options (through output.control()).

The first step is to define a grid of prediction locations.

```
parana.gr <- pred_grid(parana$borders, by=15)
points(parana)
points(parana.gr, pch=19, col=2, cex=0.25)
parana.gr0 <- locations.inside(parana.gr, parana$borders)
points(parana.gr0, pch=19, col=4, cex=0.25)
```



The following kriging call corresponds to the universal kriging. The arguments trend.d and trend.l require the specification (or terms in a formula) on both, data and prediction locations, respectively.

```
args(krige.control)

## function (type.krige = "ok", trend.d = "cte", trend.l = "cte",
##     obj.model = NULL, beta, cov.model, cov.pars, kappa, nugget,
##     micro.scale = 0, dist.epsilon = 1e-10, aniso.pars, lambda)
## NULL

args(output.control)

## function (n.posterior, n.predictive, moments, n.back.moments,
##     simulations.predictive, mean.var, quantile, threshold, sim.means,
##     sim.vars, signal, messages)
## NULL
```

```
KC <- krige.control(obj.m = parana.ml1, trend.d="1st", trend.l="1st")
OC <- output.control(simulations=TRUE, n.pred=1000,
             quantile=c(0.10, 0.25, 0.5, 0.75, 0.90), threshold = 350)
parana.kc <- krige.conv(parana, loc=parana.gr, krige = KC, output = OC)
```

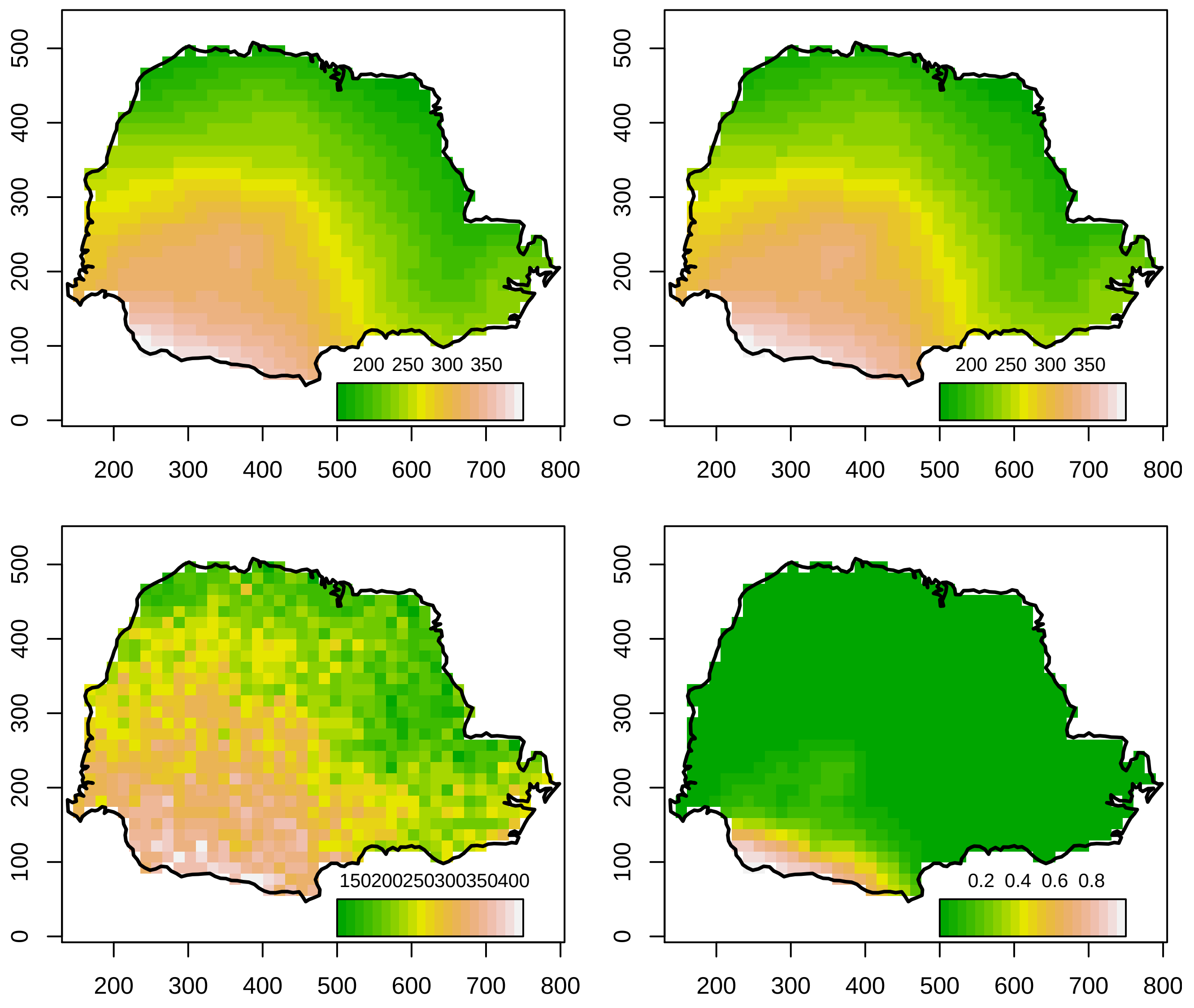
```
## krige.conv: results will be returned only for prediction locations inside the borders
## krige.conv: model with mean given by a 1st order polynomial on the coordinates
## krige.conv: sampling from the predictive distribution (conditional simulations)
## krige.conv: Kriging performed using global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

```
names(parana.kc)
```

```
##  [1] "predict"                    "krige.var"
##  [3] "beta.est"                   "distribution"
##  [5] "simulations"                "mean.simulations"
##  [7] "variance.simulations"       "quantiles.simulations"
##  [9] "probabilities.simulations" "sim.means"
## [11] ".Random.seed"               "message"
## [13] "call"
```

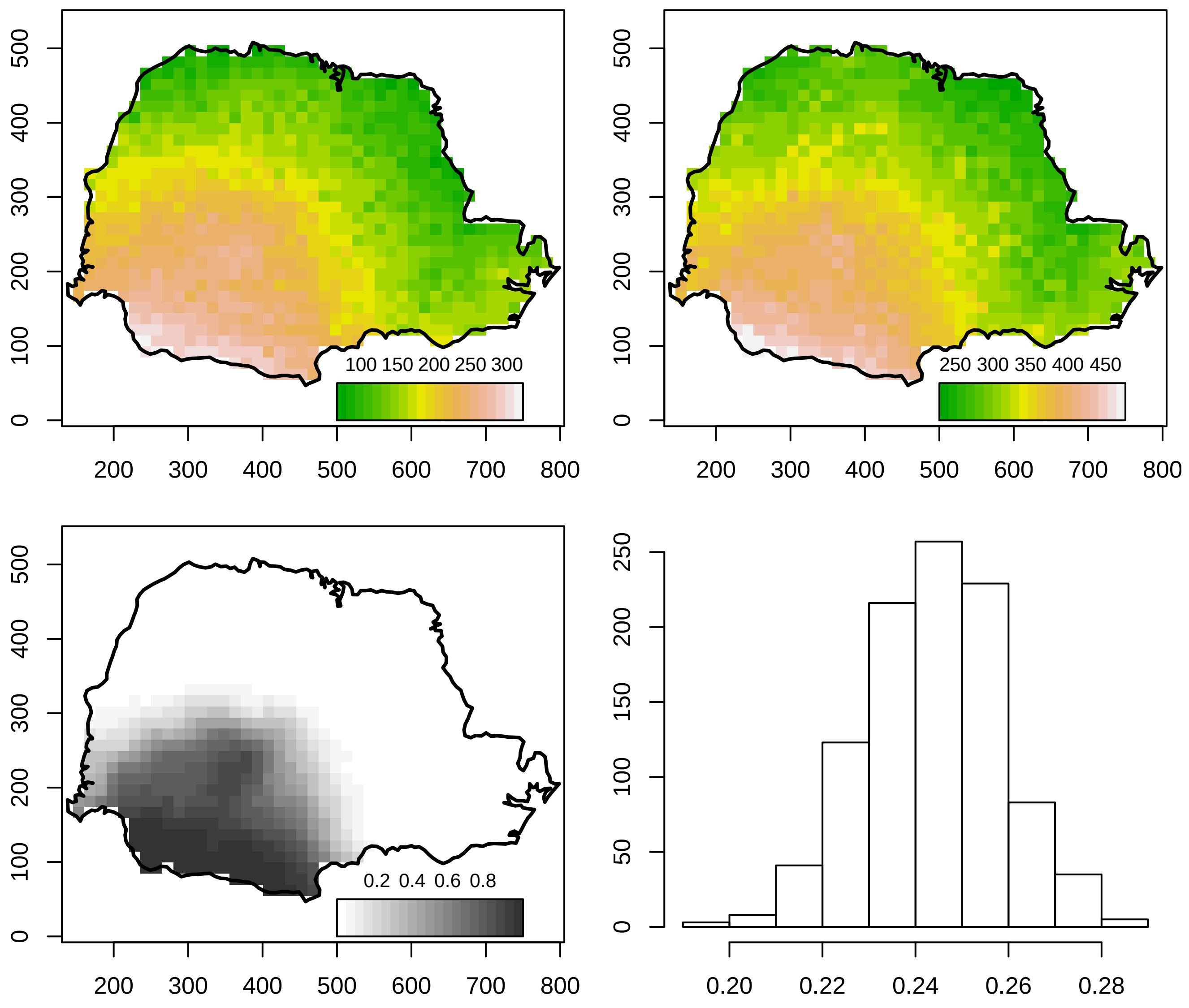
From the output provided we picked the following in the next plot: prediction means, medians, a simulation from the (joint) predictive distribution, and the probability of exceeding the value of 350.

```
par(mfrow=c(2,2), mar=c(3,3,0.5, 0.5))
image(parana.kc, col = terrain.colors(21), x.leg=c(500, 750), y.leg=c(0, 50))
image(parana.kc, val = parana.kc$quantile[,3], col=terrain.colors(21),
    x.leg=c(500, 750), y.leg=c(0, 50))
image(parana.kc, val = parana.kc$simulation[,1], col=terrain.colors(21),
    x.leg=c(500, 750), y.leg=c(0, 50))
image(parana.kc, val = 1-parana.kc$prob, col=terrain.colors(21),
    x.leg=c(500, 750), y.leg=c(0, 50))
```



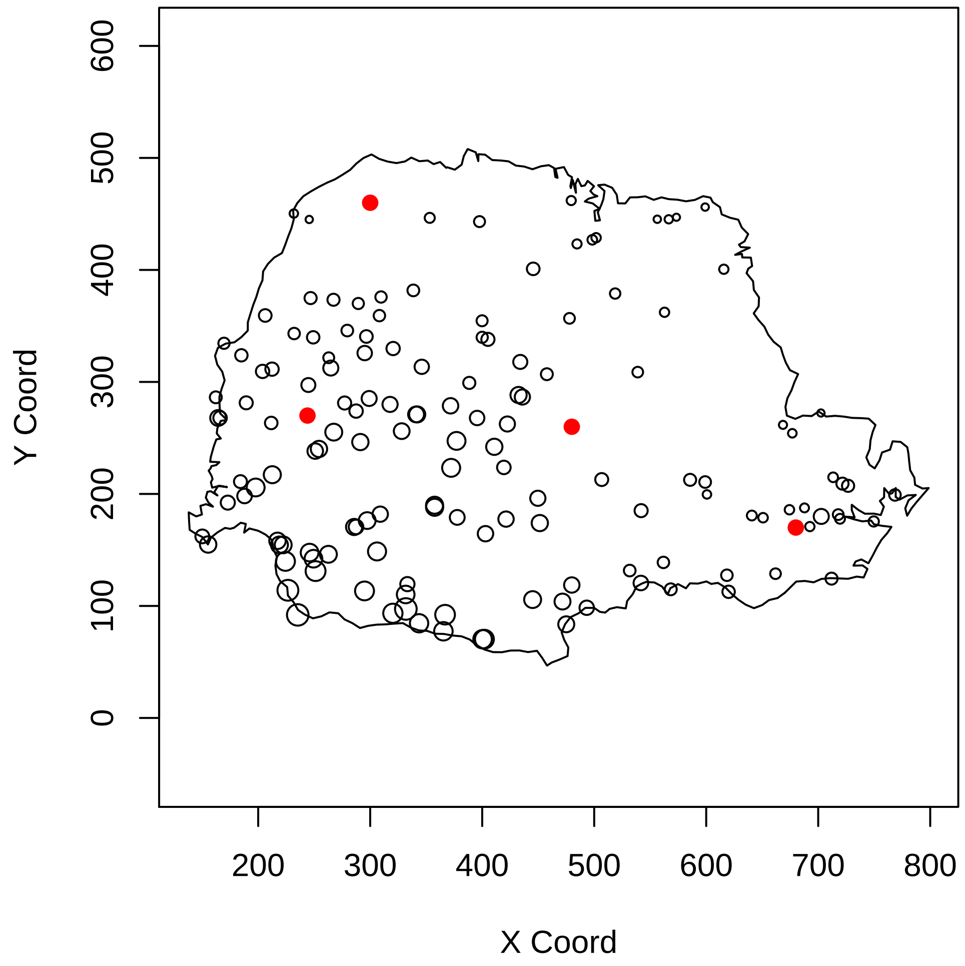
More generally, simulations can be used to derive maps of quantities of interest. The top panels in next example calls extract minimum and maximum simulated values at each location. The bottom panels shows probabilities of exceeding 300 at each location, and predictive distribution of the proportion of the area exceeding 300.

```
par(mfrow=c(2,2), mar=c(3,3,0.5, 0.5))
image(parana.kc, val = apply(parana.kc$simulation, 1, min), col=terrain.colors(21),
    x.leg=c(500, 750), y.leg=c(0, 50))
image(parana.kc, val = apply(parana.kc$simulation, 1, max), col=terrain.colors(21),
    x.leg=c(500, 750), y.leg=c(0, 50))
image(parana.kc, val=apply(parana.kc$simulations, 1, function(x) mean(x > 300, na.rm=T)),
                 x.leg=c(500, 750), y.leg=c(0,50), col=gray(seq(1, 0.2, length=21)))
hist(apply(parana.kc$simulations, 2, function(x) mean(x > 300, na.rm=T)), main="")
```

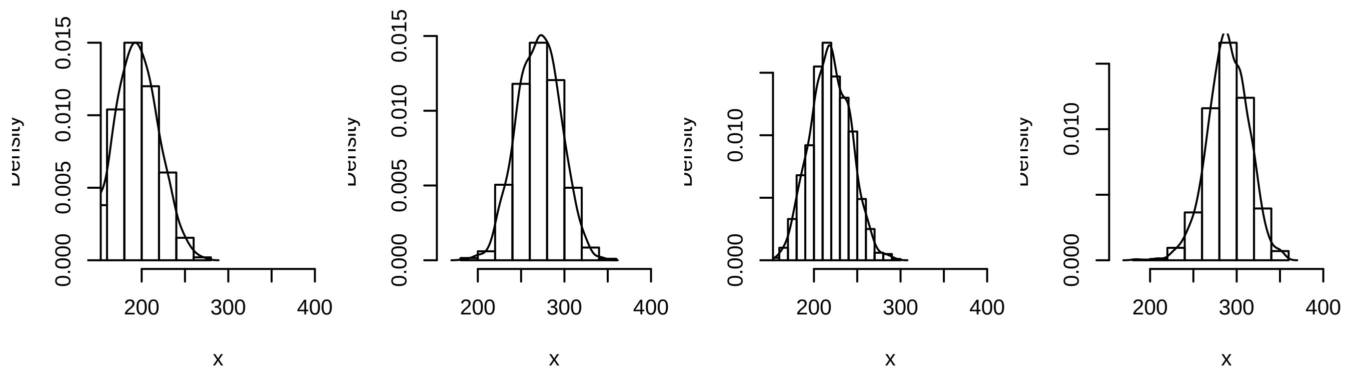


We now pick four particular locations for prediction

```
loc4 <- cbind(c(300, 480, 680, 244), c(460, 260, 170, 270))
parana.kc4 <- krige.conv(parana, loc=loc4, 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

points(parana)
points(loc4, col=2, pch=19)
```



```r
par(mfrow=c(1,4), mar=c(3.5,3.5, 0.5,0.5))
apply(parana.kc4$simulation, 1,
      function(x){ hist(x, prob=T, xlim=range(parana$data), main=""); lines(density(x))})
```



```
## NULL
```

4 Bayesian analysis using `geoR`

For the Bayesian analysis both, inference on model parameters and spatial predictions (if prediction locations are provided) are performed with a call to the function krige.bayes(). In the next call we run an analysis setting a discrete prior for the parameter in the (exponential) correlation function and fixing the nugget parameter to zero. More general call can set a discrete prior also for the relative nugget through the tausq.rel parameter.

```
args(krige.bayes)

## function (geodata, coords = geodata$coords, data = geodata$data,
##     locations = "no", borders, model, prior, output)
## NULL
```

```
parana.bayes <- krige.bayes(parana, loc=parana.gr,
                            model = model.control(trend.d="1st", trend.l="1st"),
               prior= prior.control(phi.prior="rec", phi.disc=seq(0,150, by=15)),
               output = OC)
```

```
## krige.bayes: results will be returned only for prediction locations inside the borders
## krige.bayes: model with mean given by a 1st order polynomial on the coordinates
## krige.bayes: computing the discrete posterior of phi/tausq.rel
## krige.bayes: computing the posterior probabilities.
##              Number of parameter sets:  11
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,
##
## krige.bayes: sampling from posterior distribution
## krige.bayes: sample from the (joint) posterior of phi and tausq.rel
##            2   3  4  5  6
## phi       15  30 45 60 75
## tausq.rel  0   0  0  0  0
## frequency 61 857 74  7  1
##
## krige.bayes: starting prediction at the provided locations
## krige.bayes: phi/tausq.rel samples for the predictive are same as for the posterior
## krige.bayes: computing moments of the predictive distribution
## krige.bayes: sampling from the predictive
##              Number of parameter sets:  5
## 1, 2, 3, 4, 5,
## krige.bayes: preparing summaries of the predictive distribution
```

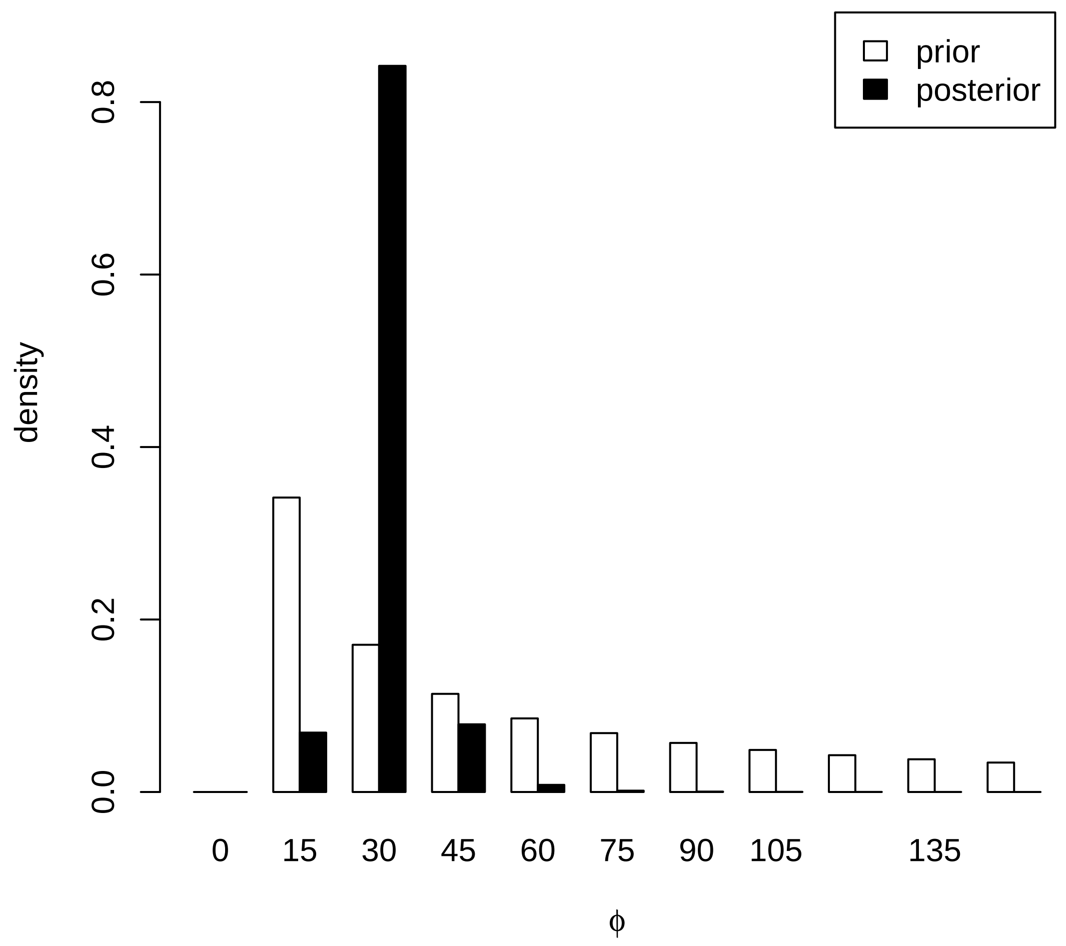
```
names(parana.bayes)
```

```
## [1] "posterior"    "predictive"   "prior"        "model"
## [5] ".Random.seed" "max.dist"     "call"
```

```
names(parana.bayes$posterior)
```

```
## [1] "beta"                "sigmasq"             "phi"
## [4] "tausq.rel"           "joint.phi.tausq.rel" "sample"
```

```
plot(parana.bayes)
```



```
## parameter `tausq.rel` is fixed
```

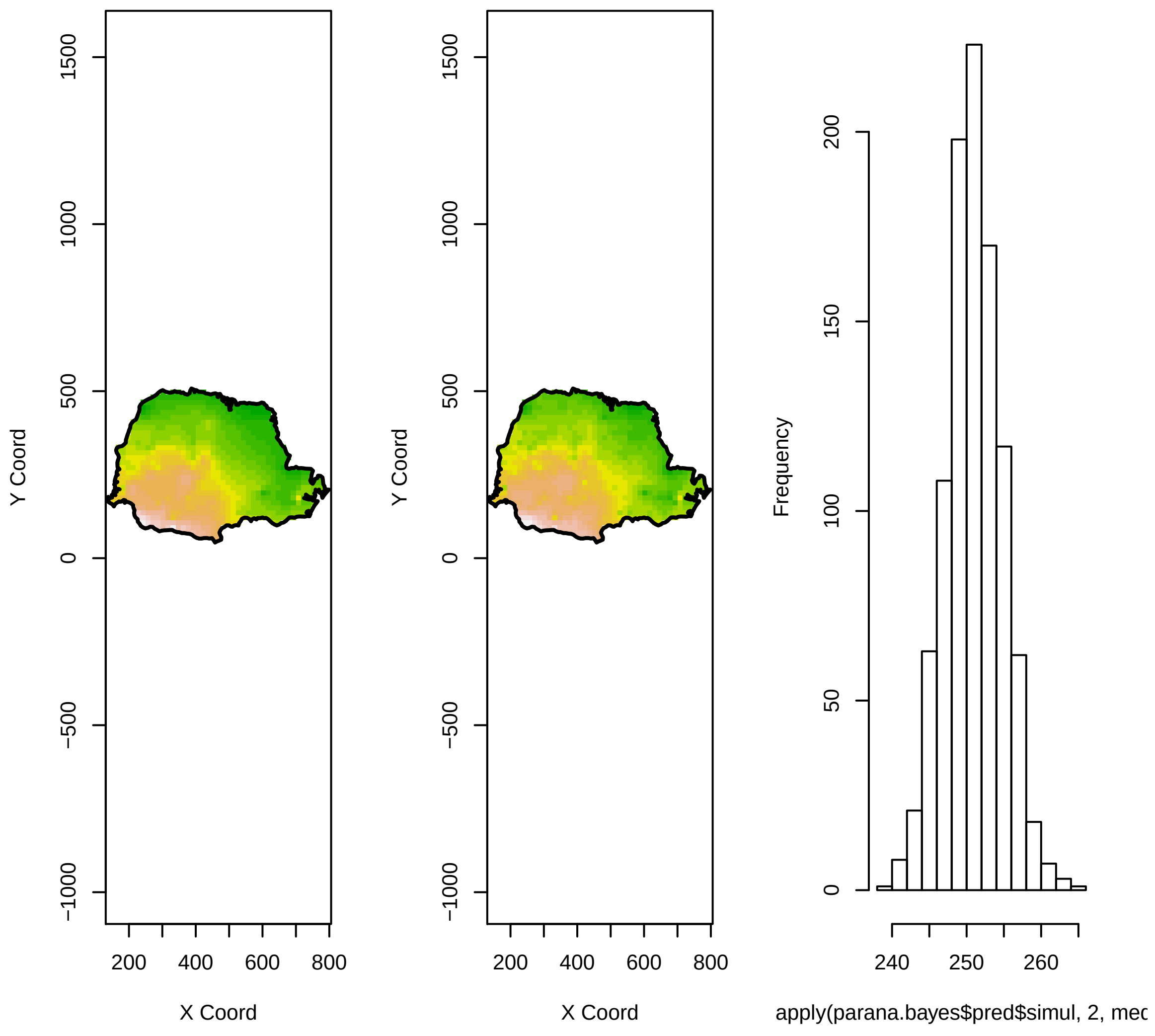
```
names(parana.bayes$predictive)
```

```
## [1] "mean"                        "variance"
## [3] "distribution"                "simulations"
## [5] "mean.simulations"            "variance.simulations"
## [7] "quantiles.simulations"       "probabilities.simulations"
## [9] "sim.means"
```

```
par(mfrow=c(1,3))
image(parana.bayes, col=terrain.colors(21))
```

```
## mapping the means of the predictive distribution
```

```
image(parana.bayes, val = apply(parana.bayes$pred$simul, 1, quantile, prob=0.90),
      col=terrain.colors(21))

      hist(apply(parana.bayes$pred$simul, 2, median), main="")
```


Y Coord
X Coord
Y Coord
X Coord
Frequency
apply(parana.bayes$pred$simul, 2, mec