Kriging

Peter Claussen

9/5/2017

Libraries

automap : Automatic interpolation package

```
library(automap)
```

```
## Loading required package: sp
```

```
## Warning: package 'sp' was built under R version 3.3.2
```

```
library(gstat)
```

```
## Warning: package 'gstat' was built under R version 3.3.2
```

```
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
## --------------------------------------------------------------
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

```
cbPalette <- c("#999999", "#E69F00", "#56B4E9", "#009E73", "#0072B2", "#D55E00", "#F0E442","#CC79A7","#
```

Data

```
load(file="sample.dat.Rda")
global.cols <- c(which(names(sample.dat)=="Latitude"), which(names(sample.dat)=="Longitude"))
metric.cols <- c(which(names(sample.dat)=="LatM"), which(names(sample.dat)=="LonM"))
yield.col <- which(names(sample.dat)=="Yield")
```

Introduction

Suppose we have a collection of spatial data, sampled repeatedly and at random over the same space at different times - for example, yield maps for the same field from different harvest seasons. The random nature of sampling for each year means that it is very unlikely that we've measured yield at the same exact point in space for more than one year. How do we combine these data?

One method is to define a grid of fixed points, sufficiently spaced to suit our needs, but no more closely spaced than our original data, and for simplicity usually evenly spaced. We will almost certainly not have observed an actual yield at each of these (unobserved) locations, so we use observations at random locations to impute values at our fixed points.

Kriging is a form of interpolation, specifically in the ordinary case, a linear interpolation. We write a linear interpolator of the form

$$\hat{Y}(x) = \Sigma_{i=1}^{n} \theta_i Y(x_i)$$

We typically use the contrainst

$$\Sigma_{i=1}^{n} \theta_i = 1$$

Why? Suppose we let $\theta_i = 1/n$ for all i. Then

$$\hat{Y}(x) = \Sigma_{i=1}^{n} (\frac{1}{n}) Y(x_i) = \frac{\Sigma_{i=1}^{n} Y(x_i)}{n}$$

which is our usual estimate of the mean of Y when Y_i are uncorrelated. Since we expect the values to be uncorrelated, all values carry equal weight, and that weight is ditributed over all observations.

(Note here the difference between spatial interpolation and trend analysis. Using trend analysis, we use observations to estimate parameters the product an unobserved Y based on position (lat, lon) in space. With the interpolation methods here, we impute Y as a function of it's neighboring observations - in this way interpolation can be said to "honor the data". Once we've estimated $\beta_1, ..., \beta_n$ from the data, we tend to ignore the individual Y_i).

With spatial data, we don't assume that observations are uncorrelated. The simplest assumption for spatial data is that points are correlated by their distances - points near to each other are more alike than points that are distant from each other. This leads a simple interpolator, the inverse distance weight interpolator.

Suppose we wish to provide a value at point Y'. Denote the distance from Y' to each Y_i as $d(Y', Y_i)$. Then let

$$\theta_i = \frac{d(Y', Y_i)^{-p}}{\Sigma_{j=1}^{n} d(Y', Y_j)^{-p}}$$

for some $p \geq 1$. Obviously, if $p = 1$ then

$$\Sigma_{i=1}^{n} \theta_i = \Sigma_{i=1}^{n} \frac{d(Y', Y_i)^{-p}}{\Sigma_{j=1}^{n} d(Y', Y_j)^{-p}} = \frac{\Sigma_{i=1}^{n} d(Y', Y_i)^{-p}}{\Sigma_{i=1}^{n} d(Y', Y_i)^{-p}} = 1$$

Note that this form assumes that correlation between points is perfectly correlated with distance - the weight is only dependent on actual distance. We have seen from the variograms, though, that observations are not perfectly correlated in space. (If they were, there would be no nugget in the variogram - points at arbitrarily closed distances would be arbitrarily close to equal).

Instead, we use informnation from the variogram to compute weights. This process is termed Kriging, after the South African mining engineer D.G. Krige.

(Typically, this process assumes that Y is second-order stationary, and less often stated explicitly that Y are isotropic).

We determine from our variagram three parameters,

$\bar{\gamma}_0$, $\bar{\gamma}_\infty$ and $\bar{\gamma}_h$

the value of the (fitted) variogram at distance h. Then we let

$$\bar{C}(h) = \bar{\gamma}_\infty - \bar{\gamma}_0 - \bar{\gamma}_h$$

From this, we use the kriging equations

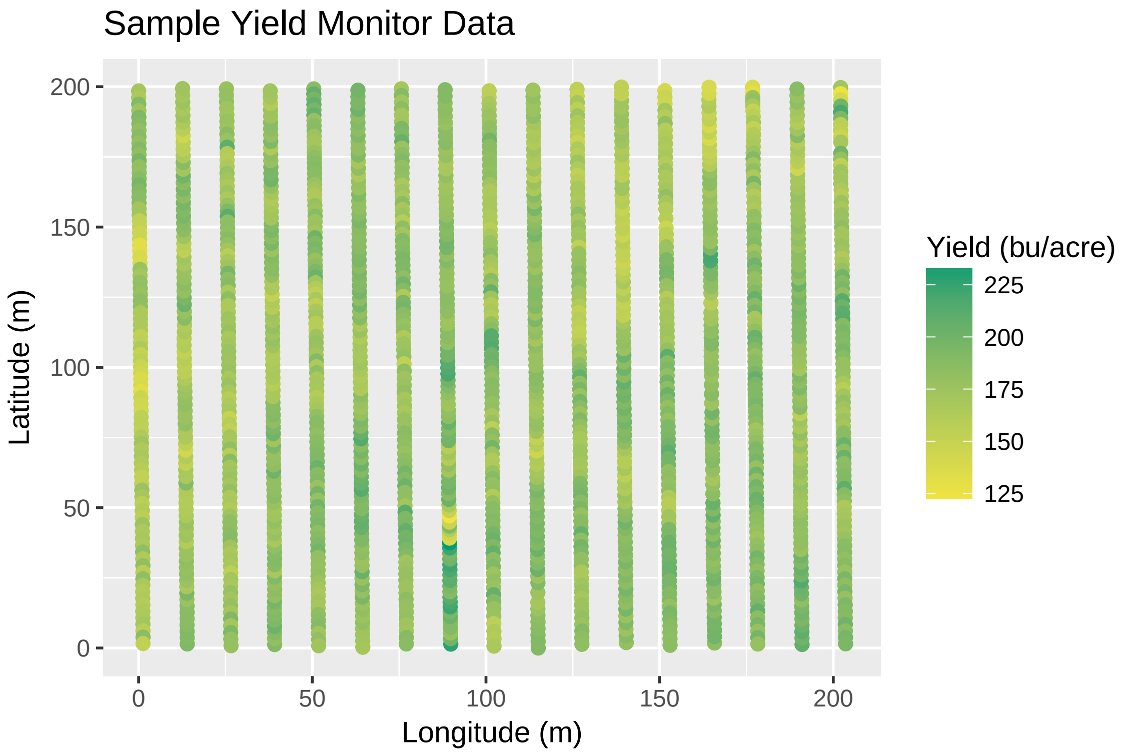
$$\Sigma_{j=1}^{n} \theta_j \bar{C_{ij}} + \psi = \bar{C_i'}$$

where $C_{ij} = \gamma(h_{ij})$ is the value of the variogram at lag h_{ij} between points Y_i and Y_j and $C_{i'} = \gamma(h_{i'})$ is the value of the variogram at lag $h_{i'}$.

That's a lot of summing over a lot of points, so in practice we limit the number of points to a neighborhood around our fixed points of interest.

Kriging in R

```
ggplot(sample.dat, aes(LonM, LatM)) +
geom_point(aes(colour = Yield),size=2) +
scale_colour_gradient(low=cbPalette[7], high=cbPalette[4]) +
labs(colour = "Yield (bu/acre)", x="Longitude (m)", y="Latitude (m)", title = "Sample Yield Monitor Dat
```



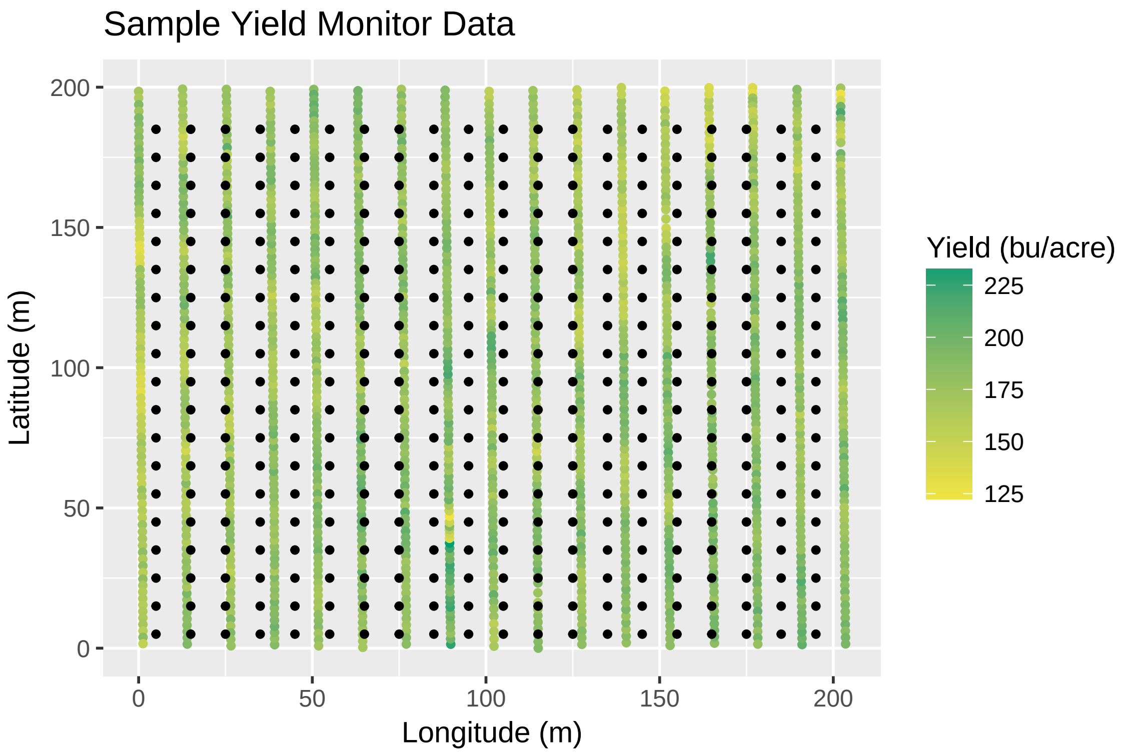
Kriging is used to interpolate values - to estimate values at unobserved locations. We'll create a uniformly spaced set of points to estimate. For simplicity, we'll choose points at 10 meter intervals, starting 5 meters from the origin.

```
colPoints <- floor(max(sample.dat$LonM)/10)
rowPoints <- floor(max(sample.dat$LatM)/10)
sample4.grd <- expand.grid(LatM=((1:rowPoints)*10)-5,
                          LonM=((1:colPoints)*10)-5)
head(sample4.grd)
```

```
##   LatM LonM
## 1    5    5
## 2   15    5
## 3   25    5
## 4   35    5
## 5   45    5
## 6   55    5
```

Superimpose these points over our yield data,

```
ggplot(sample.dat, aes(LonM, LatM)) +
geom_point(aes(colour = Yield),size=1) +
scale_colour_gradient(low=cbPalette[7], high=cbPalette[4]) +
  geom_point(aes(LonM, LatM),data=sample4.grd, size=1) +
labs(colour = "Yield (bu/acre)", x="Longitude (m)", y="Latitude (m)", title = "Sample Yield Monitor Data
```



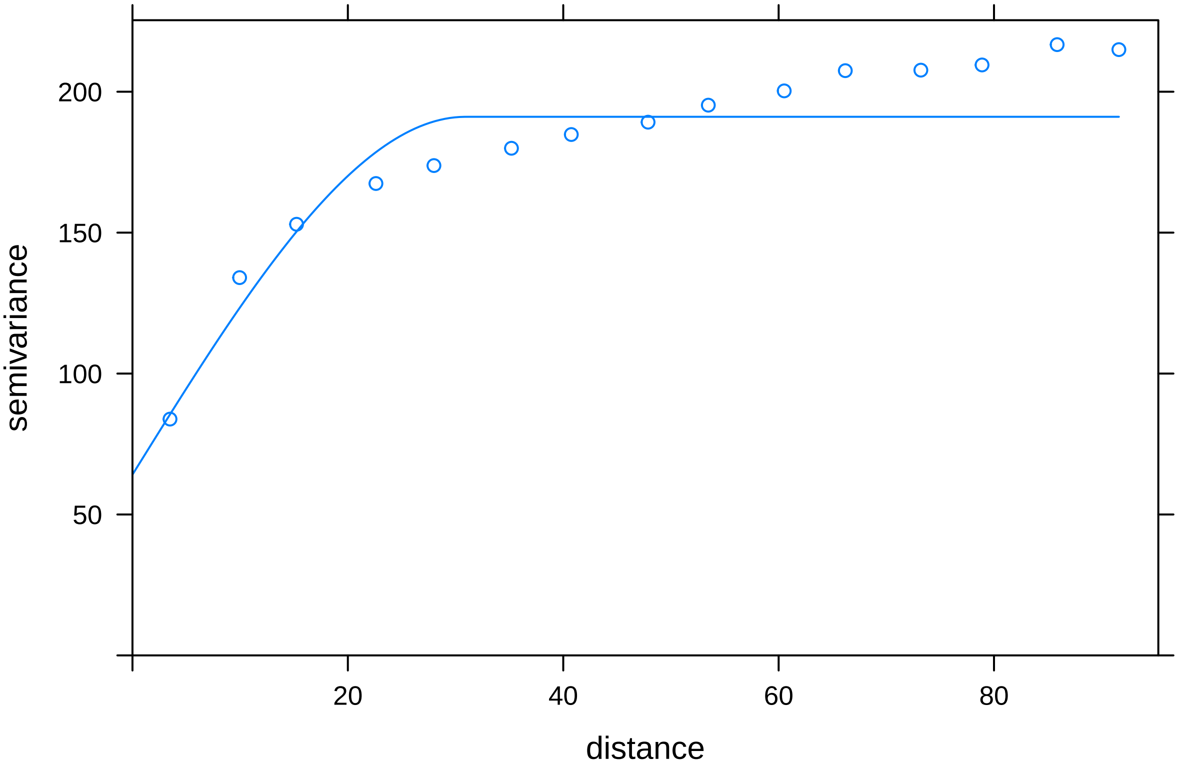
gstat

We need a variogram, so start with a spherical model from earlier,

```
Yield.var <- variogram(Yield~1,
                        locations=~LonM+LatM,
                        data=sample.dat)
print(Yield.vgm <- fit.variogram(Yield.var, vgm(250, "Sph", 80, 100)))
```

```
##   model     psill    range
## 1   Nug  64.13601  0.00000
## 2   Sph 126.97034 30.92964
```

```
plot(Yield.var, model=Yield.vgm)
```



Now use the `krig` function to project the yield data onto our sample grid.

```
Yield.krig <- krige(id="Yield",
                    formula=Yield~1,
                    data = sample.dat,
                    newdata = sample4.grd,
                    model = Yield.vgm,
                    maxdist = 100,
  locations=~LatM + LonM)
```

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

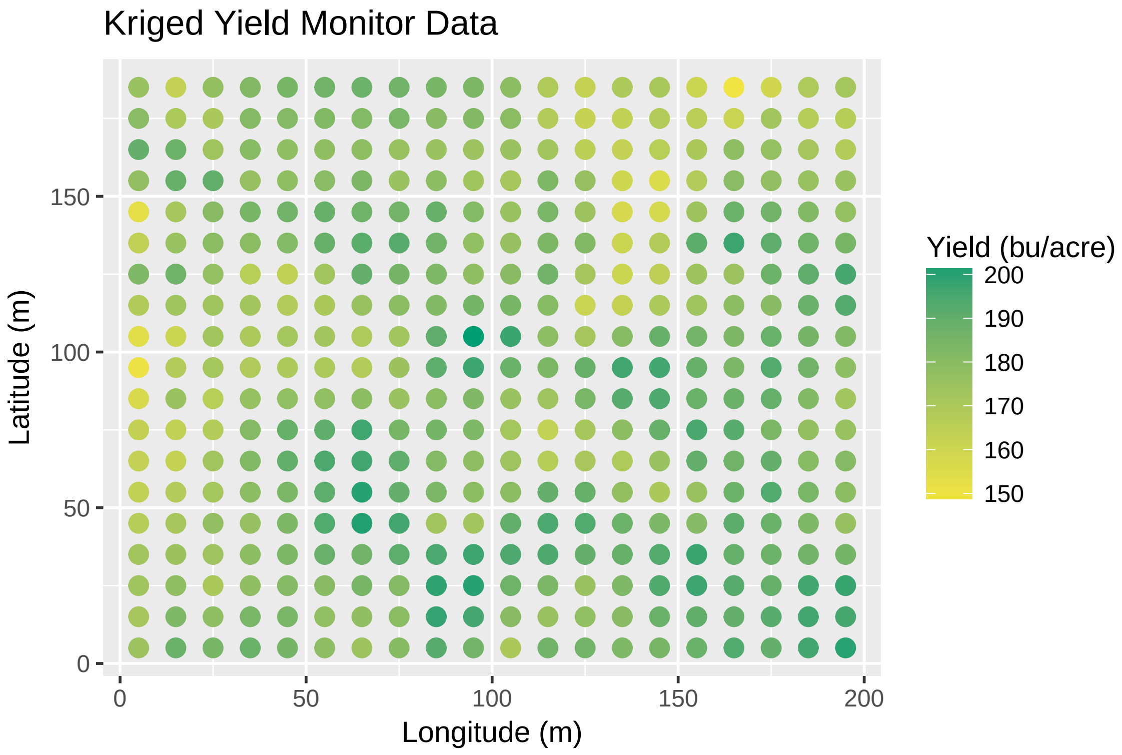
We can extract the predicted values, and plot the results

```
str(Yield.krig)
```

```
## 'data.frame':    380 obs. of  4 variables:
##  $ LatM      : num  5 15 25 35 45 55 65 75 85 95 ...
##  $ LonM      : num  5 5 5 5 5 5 5 5 5 5 ...
##  $ Yield.pred: num  175 172 173 173 167 ...
##  $ Yield.var : num  101.3 99.8 99.6 99.9 100.4 ...
```

```
sample4.grd$Yield.krig <- Yield.krig$Yield.pred
```

```
ggplot(sample4.grd, aes(LonM, LatM)) +
geom_point(aes(colour = Yield.krig),size=3) +
scale_colour_gradient(low=cbPalette[7], high=cbPalette[4]) +
labs(colour = "Yield (bu/acre)", x="Longitude (m)", y="Latitude (m)", title = "Kriged Yield Monitor Data
```



geoR

There are a wide range of options for kriging in R. We explore a couple others before we move on. The next is geoR. As befor, we convert to geodata and fit to a variogram.

```
Yield.idx = which(names(sample.dat)=="Yield")
metric.col <- c(which(names(sample.dat)=="LatM"), which(names(sample.dat)=="LonM"))
Yield.gdat <- as.geodata(sample.dat, coords.col = metric.col, data.col = Yield.idx)
Yield.gvar <- variog(Yield.gdat)
```

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

```
init.cov <- expand.grid(seq(10,200,l=10), seq(0,40,l=5))
Yield.gfit <- variofit(Yield.gvar, cov.model="sph",
                       ini.cov.pars=init.cov,
                       fix.nugget=FALSE, nugget=50)
```

```
## variofit: covariance model used is spherical
## variofit: weights used: npairs
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq  phi   tausq kappa
## initial.value "157.78" "40"  "50"  "0.5"
## status        "est"    "est" "est" "fix"
## loss value: 499605387.978827
```

```
Yield.gfit
```

```
## variofit: model parameters estimated by WLS (weighted least squares):
## covariance model is: spherical
## parameter estimates:
##    tausq  sigmasq      phi
## 165.9453 108.9368 347.2211
## Practical Range with cor=0.05 for asymptotic range: 347.2211
##
## variofit: minimised weighted sum of squares = 72336853
```

```
geoR.krig <- krige.conv(Yield.gdat, locations=sample4.grd, krige=krige.control(obj.model=Yield.gfit))
```

```
## Warning in .check.locations(locations): locations provided with a matrix
## or data-frame with more than 2 columns. Only the first two columns used as
## coordinates
```

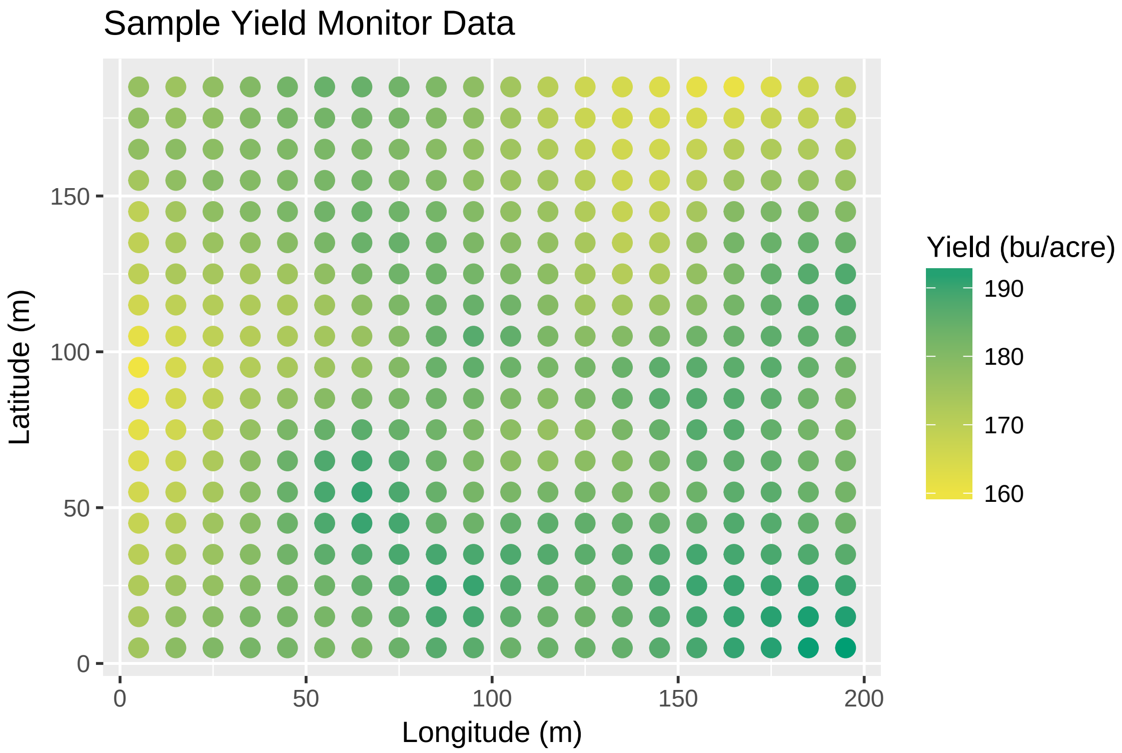
```
## krige.conv: model with constant mean
## krige.conv: Kriging performed using global neighbourhood
```

```
str(geoR.krig)
```

```
## List of 6
##  $ predict     : num [1:380] 175 174 172 171 168 ...
##  $ krige.var   : num [1:380] 176 174 174 174 174 ...
##  $ beta.est    : Named num 174
##   ..- attr(*, "names")= chr "beta"
##  $ distribution: chr "normal"
##  $ message     : chr "krige.conv: Kriging performed using global neighbourhood"
##  $ call        : language krige.conv(geodata = Yield.gdat, locations = sample4.grd, krige = krige.co
##  - attr(*, "sp.dim")= chr "2d"
##  - attr(*, "prediction.locations")= symbol sample4.grd
##  - attr(*, "parent.env")=<environment: R_GlobalEnv>
##  - attr(*, "data.locations")= language Yield.gdat$coords
##  - attr(*, "class")= chr "kriging"
```

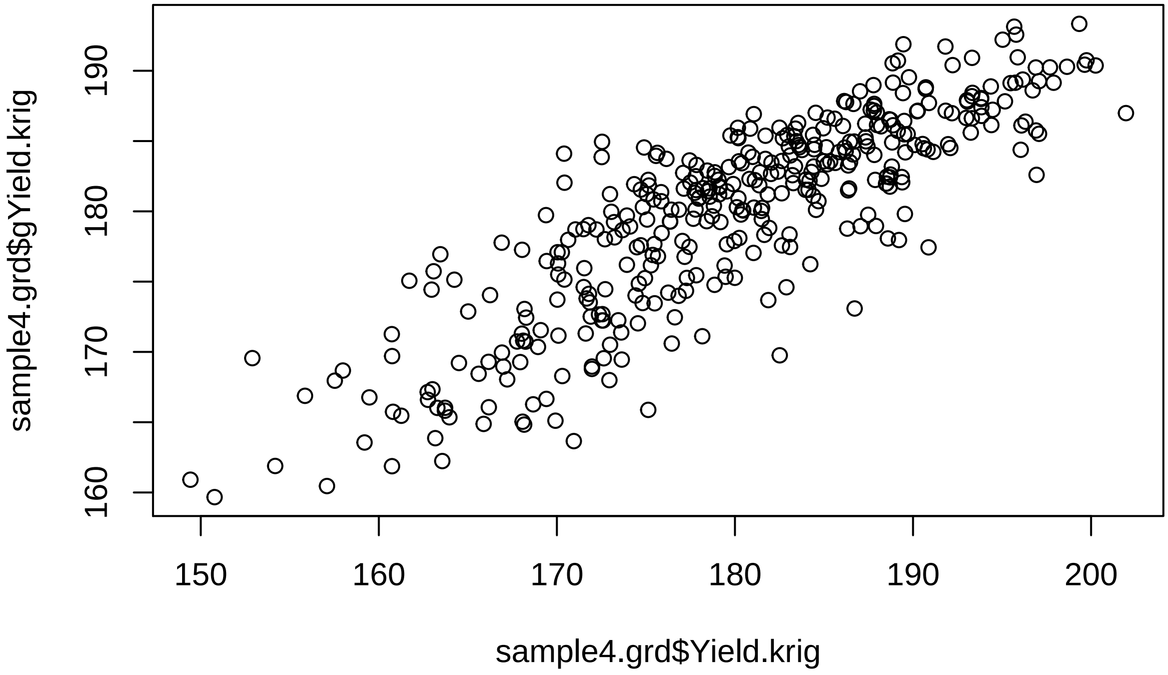
```
sample4.grd$gYield.krig <- geoR.krig$predict
```

```
ggplot(sample4.grd, aes(LonM, LatM)) +
geom_point(aes(colour = gYield.krig),size=3) +
scale_colour_gradient(low=cbPalette[7], high=cbPalette[4]) +
labs(colour = "Yield (bu/acre)", x="Longitude (m)", y="Latitude (m)", title = "Sample Yield Monitor Data
```



We can compare the two methods,

```
plot(sample4.grd$Yield.krig,sample4.grd$gYield.krig)
```



I can get this to work if I run just `geoR` code in R, but not from within this typeset document; I'm not sure what's causing the conflict.

```
par(mfrow=c(1,3))
image(geoR.krig)
contour(geoR.krig)
```

```
persp(geoR.krig)
```

automap

Automap produces interesting graphics. Note that `gstat` and `geoR` can be used to similarly produce graphics, the advantage of autoKrig is to simplify fitting.

autoKrig

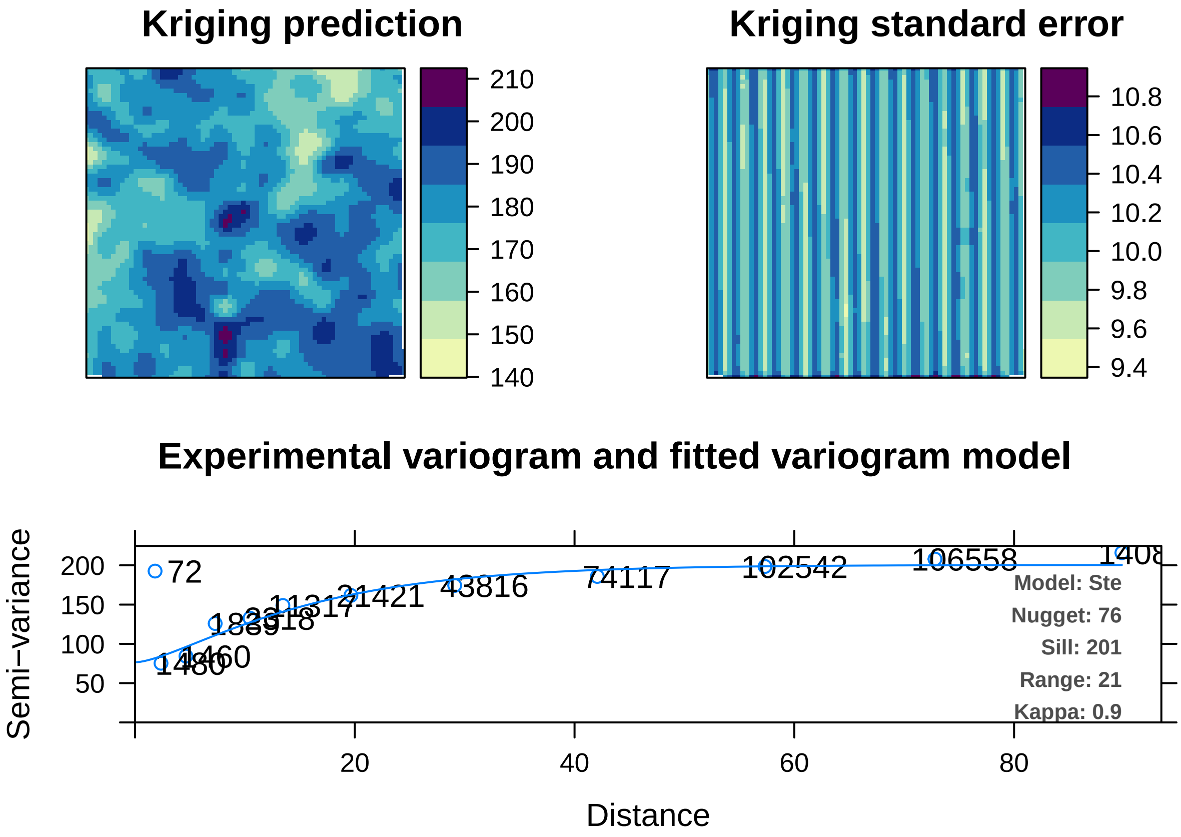
```r
coordinates(sample.dat) = ~ LonM+LatM

# Ordinary kriging, no new_data object
Yield.autokrig = autoKrige(Yield~1, sample.dat)
```

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

```r
str(Yield.autokrig)
```

```
## List of 4
##  $ krige_output:Formal class 'SpatialPixelsDataFrame' [package "sp"] with 7 slots
##   .. ..@ data       :'data.frame':   4969 obs. of  3 variables:
##   .. .. ..$ var1.pred : num [1:4969] 186 187 186 185 183 ...
##   .. .. ..$ var1.var  : num [1:4969] 104 108 115 114 105 ...
##   .. .. ..$ var1.stdev: num [1:4969] 10.2 10.4 10.7 10.7 10.3 ...
##   .. ..@ coords.nrs : int [1:2] 1 2
##   .. ..@ grid       :Formal class 'GridTopology' [package "sp"] with 3 slots
##   .. .. .. ..@ cellcentre.offset: Named num [1:2] 1.95 1.3
##   .. .. .. .. ..- attr(*, "names")= chr [1:2] "x1" "x2"
##   .. .. .. ..@ cellsize         : Named num [1:2] 2.84 2.84
##   .. .. .. .. ..- attr(*, "names")= chr [1:2] "x1" "x2"
##   .. .. .. ..@ cells.dim        : Named int [1:2] 72 70
##   .. .. .. .. ..- attr(*, "names")= chr [1:2] "x1" "x2"
##   .. ..@ grid.index : int [1:4969] 4973 4974 4975 4976 4977 4978 4979 4980 4981 4982 ...
##   .. ..@ coords     : num [1:4969, 1:2] 13.3 16.1 19 21.8 24.7 ...
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : NULL
##   .. .. .. ..$ : chr [1:2] "x1" "x2"
##   .. ..@ bbox       : num [1:2, 1:2] 1.95 1.3 203.41 197.08
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:2] "x1" "x2"
##   .. .. .. ..$ : chr [1:2] "min" "max"
##   .. ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slot
##   .. .. .. ..@ projargs: chr NA
##  $ exp_var     :Classes 'gstatVariogram' and 'data.frame':   12 obs. of  6 variables:
##   ..$ np     : num [1:12] 72 1480 1460 1839 2318 ...
##   ..$ dist   : num [1:12] 1.82 2.36 4.63 7.29 10.46 ...
##   ..$ gamma  : num [1:12] 192.7 75.3 84.4 125.9 132.7 ...
##   ..$ dir.hor: num [1:12] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ dir.ver: num [1:12] 0 0 0 0 0 0 0 0 0 0 ...
##   ..$ id     : Factor w/ 1 level "var1": 1 1 1 1 1 1 1 1 1 1 ...
##   ..- attr(*, "direct")='data.frame':    1 obs. of  2 variables:
##   .. ..$ id       : Factor w/ 1 level "var1": 1
##   .. ..$ is.direct: logi TRUE
##   ..- attr(*, "boundaries")= num [1:12] 2 3.99 5.99 8.99 11.98 ...
##   ..- attr(*, "pseudo")= num 0
##   ..- attr(*, "what")= chr "semivariance"
##  $ var_model   :Classes 'variogramModel' and 'data.frame':   2 obs. of  9 variables:
##   ..$ model: Factor w/ 20 levels "Nug","Exp","Sph",..: 1 7
##   ..$ psill: num [1:2] 76.5 124.1
##   ..$ range: num [1:2] 0 21.1
##   ..$ kappa: num [1:2] 0 0.9
##   ..$ ang1 : num [1:2] 0 0
##   ..$ ang2 : num [1:2] 0 0
##   ..$ ang3 : num [1:2] 0 0
##   ..$ anis1: num [1:2] 1 1
##   ..$ anis2: num [1:2] 1 1
##   ..- attr(*, "singular")= logi FALSE
##   ..- attr(*, "SSErr")= num 320541
##   ..- attr(*, "call")= language fit.variogram(object = experimental_variogram, model = vgm(psill = p
##  $ sserr       : num 320541
##  - attr(*, "class")= chr [1:2] "autoKrige" "list"
```

```
plot(Yield.autokrig)
```



Variations on kriging

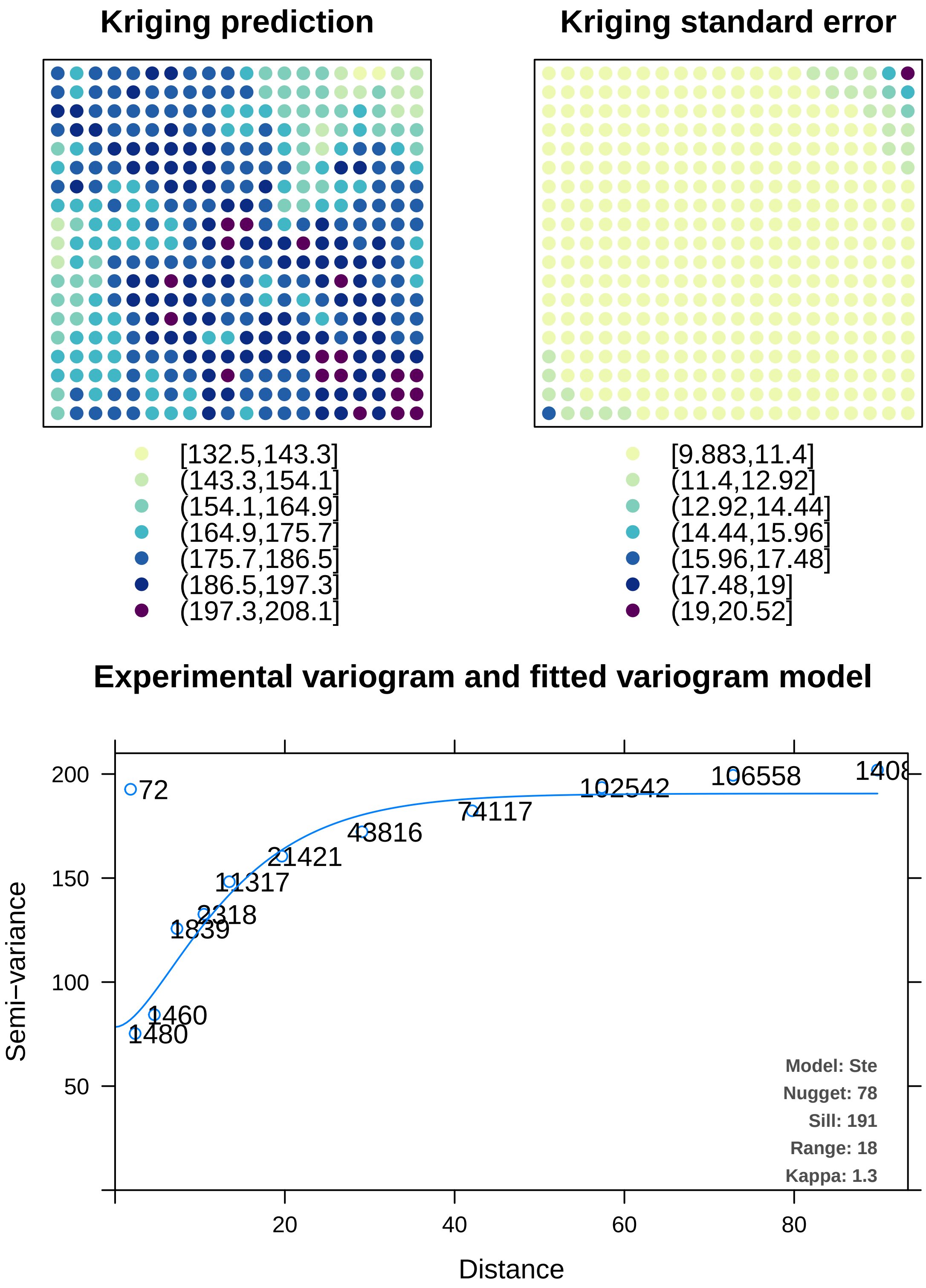
Universal kriging

Universal kriging assumes an underlying deterministic model. For example, we might include a polynomial surface trend by

```
coordinates(sample4.grd) = ~ LonM+LatM
Yield.ukrig = autoKrige(Yield~poly(LatM + LonM, 7), input_data=sample.dat, new_data=sample4.grd)
```

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

```
#with this, predictors are required for new locations.
#Yield.ukrig = autoKrige(Yield~poly(LatM + LonM, 7), input_data=sample.dat)
plot(Yield.ukrig)
```

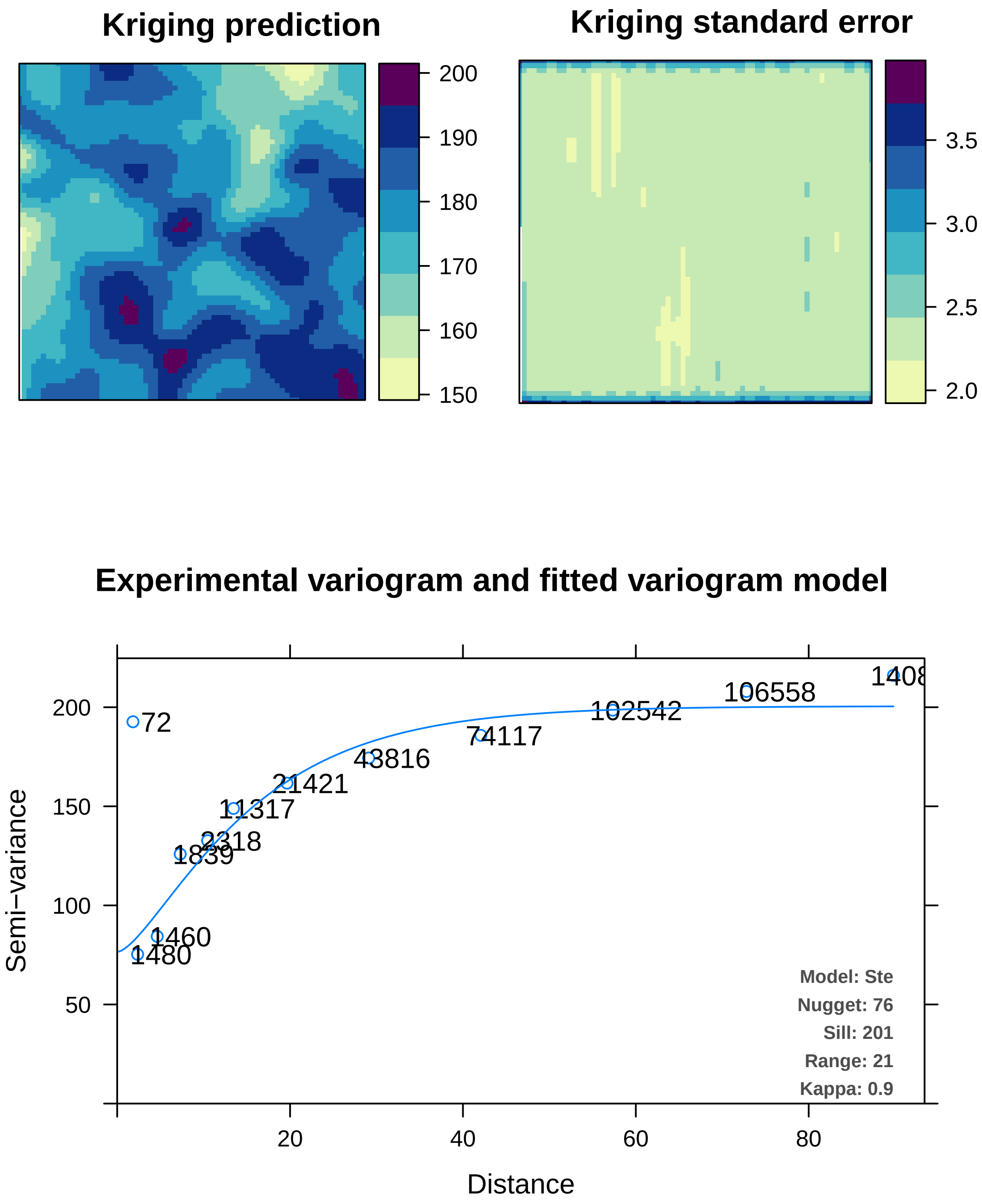


Block kriging

```
Yield.blockkrig = autoKrige(Yield~1, sample.dat, block = c(20,20))
```

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

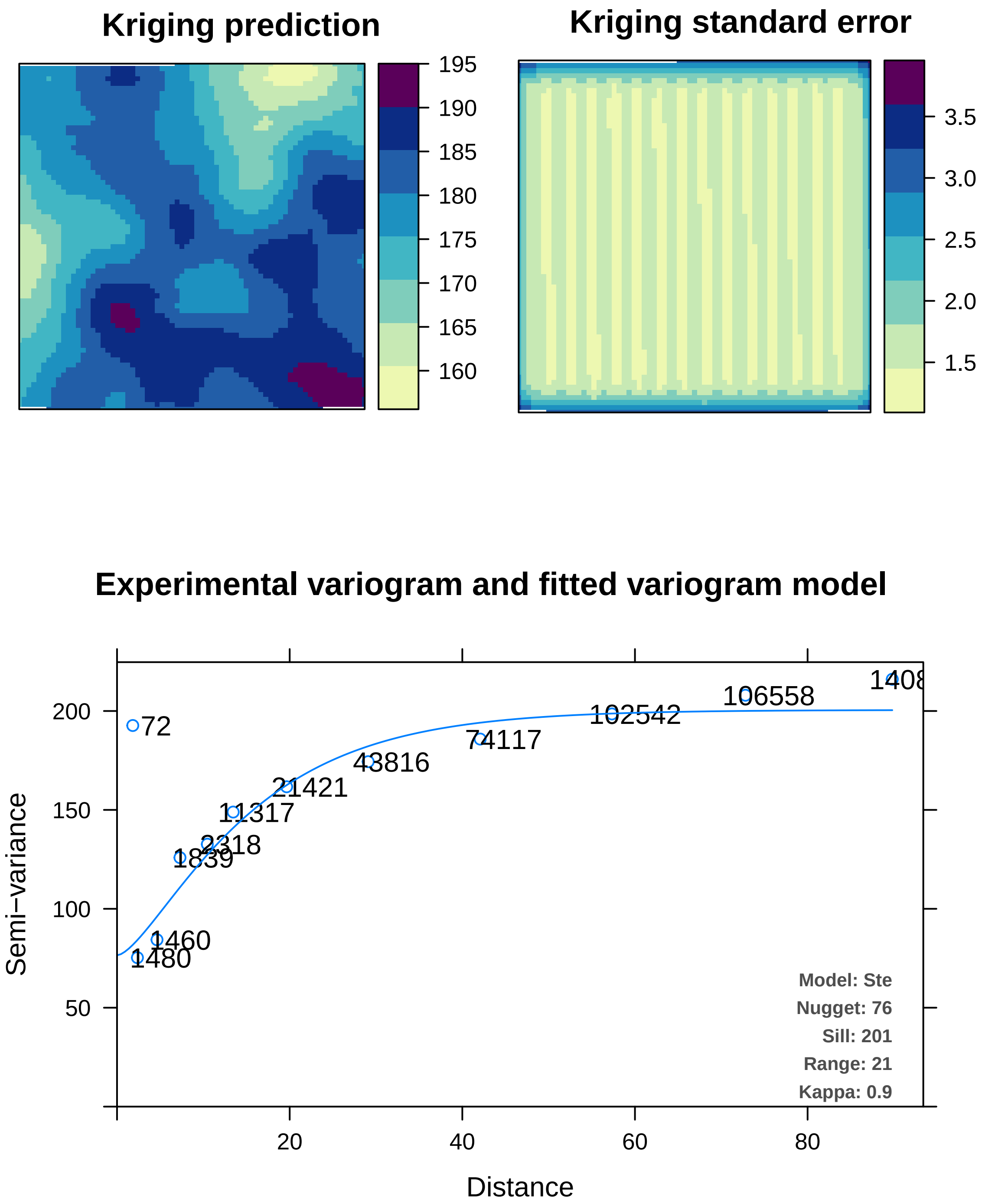
```r
plot(Yield.blockkrig)
```



```r
Yield.blockkrig = autoKrige(Yield~1, sample.dat, block = c(40,40))
```

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

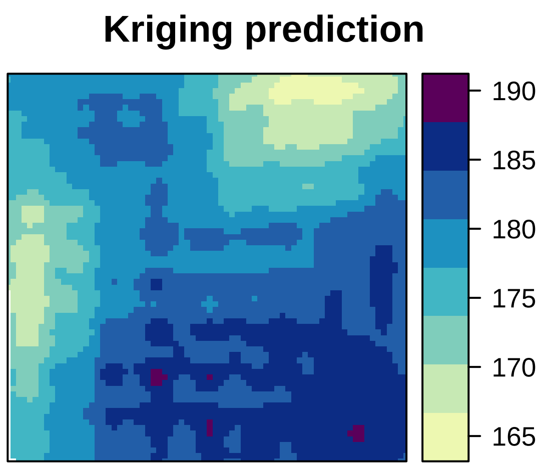
```r
plot(Yield.blockkrig)
```

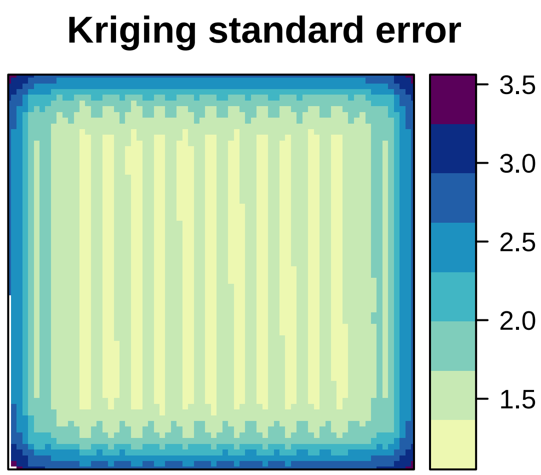


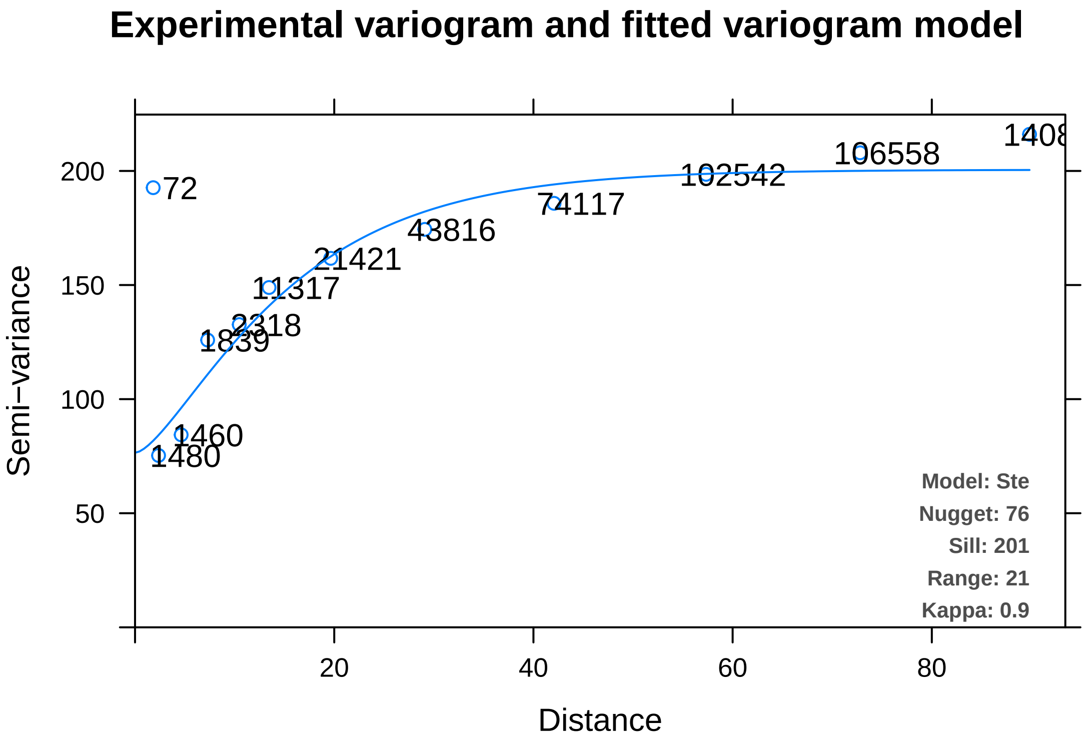
```
Yield.blockkrig = autoKrige(Yield~1, sample.dat, block = c(80,80))
```

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

```
plot(Yield.blockkrig)
```


Kriging prediction
190
185
180
175
170
165


Kriging standard error
3.5
3.0
2.5
2.0
1.5


Experimental variogram and fitted variogram model
Semi−variance
200
150
100
50
Distance
20
40
60
80
72
1480
1460
1839
2318
11317
21421
43816
74117
102542
106558
1408
Model: Ste
Nugget: 76
Sill: 201
Range: 21
Kappa: 0.9