

Assignment 3: Computer Lab

Sungil Kim

Contents

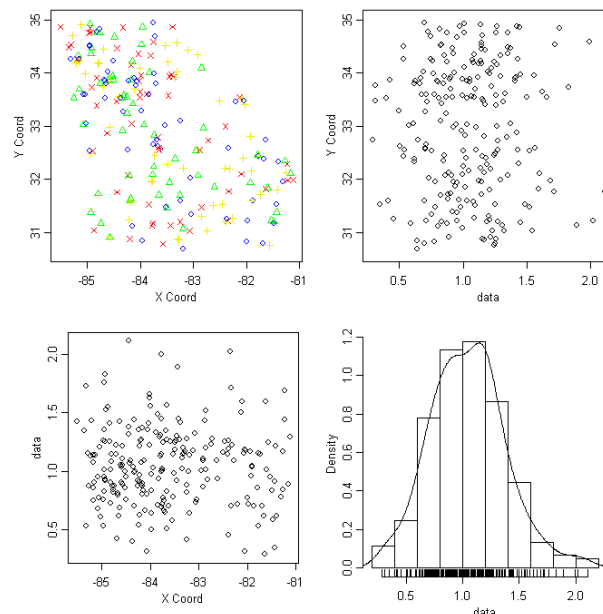
1. Data Description
2. Variogram
 - 2.1 Empirical variogram estimation
 - 2.2 Valid model selection and Model fitting
3. Spatial Prediction: Kriging
 - 3.1 Without Demo. and Competition Data
 - 3.2 Variable Selection (stepwise regression)
 - 3.3 With Demo. and Competition Data
4. Appendix

1. Data Description

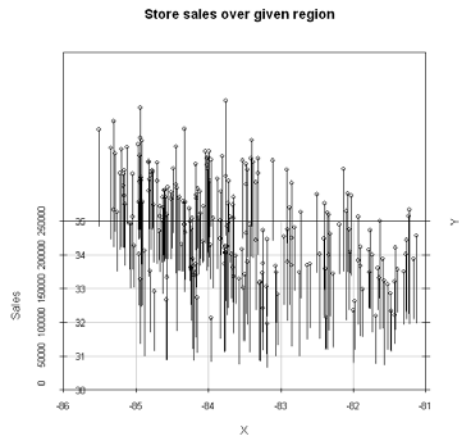
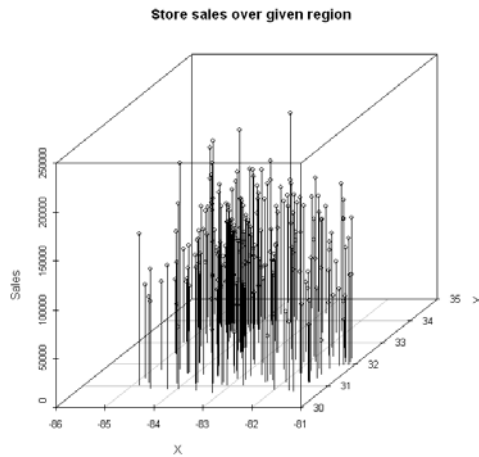
'store' contains

1. Simulated sales data for 2000
2. Demographics data
3. Competition data
4. Latitude and longitude of the store locations

'store.att' consists of the coordinate of 225 stores and their sales data in 2000. For convenient, the sales values are divided by 100,000. We assume that the sales data, $\{Y(s), s \text{ in } R\}$ a spatial process. The process is homogeneous, which means intrinsically stationary and isotropic.



According to density function, it does not skewed, so transformation is not considered.

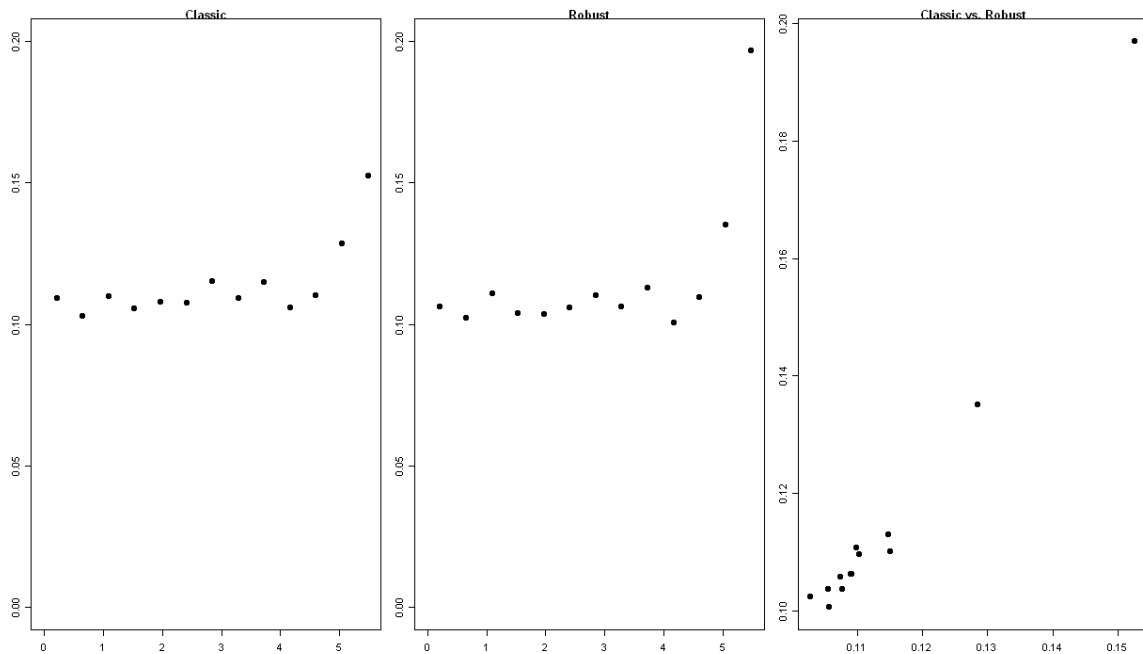


2. Variogram

2.1 Empirical variogram estimation

Here, the empirical variogram for 2000 sales are estimated in two way, classical and robust models, and are compared.

```
Par(mfrow=c(1,3),mar=c(2,2,1,1))
v1 <- variog(□ords=store.att$coords,data=store.att$data)
plot(v1,main="Classic",lwd=3,ylim=c(0,0.2))
v2<-variog(□ords=store.att$coords,data=store.att$data,estimator.type = c("modulus"))
plot(v2,main="Robust",lwd=3,ylim=c(0,0.2))
plot(v1$v,v2$v,lwd=3,main="Classic vs. Robust")
```



Classic vs. Robust pairwise plots shows that the normality assumption is valid.

2.2 Valid Model Selection and Model Fitting

There are many examples of covariogram models (see lecture note, p 4)

- Exponential
- Gaussian
- Powered exponential
- Spherical
- Cauchy
- Circular
- Cubic
- Matern
- Power.exponential
- gneiting
- gneiting.matern
- Power
- Wave...

Each model's parameters are estimated by OLS(ordinary least squares) and R gives minimized sum of squares. I choose 'Cauchy' model, which gives the smallest minimized sum of squares among above candidates.

```
fit1b <- variofit(v1,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
                 weights='equal',minimisation.function='optim')
> fit1b
variofit: model parameters estimated by OLS (ordinary least squares):
covariance model is: cauchy with fixed kappa = 0.5
parameter estimates:
   tausq  sigmasq    phi
0.1029  78.6590 197.8559

variofit: minimised sum of squares = 9e-04
```

In addition, estimated parameters for σ^2 and ϕ are

```
> fit1b$cov.pars
[1] 78.65897 197.85591
```

'Cauchy' Covariance Function

$$C(h) = \sigma^2 \cdot \left[1 + \left(\frac{h}{\phi} \right)^2 \right]^{(-\kappa)}$$

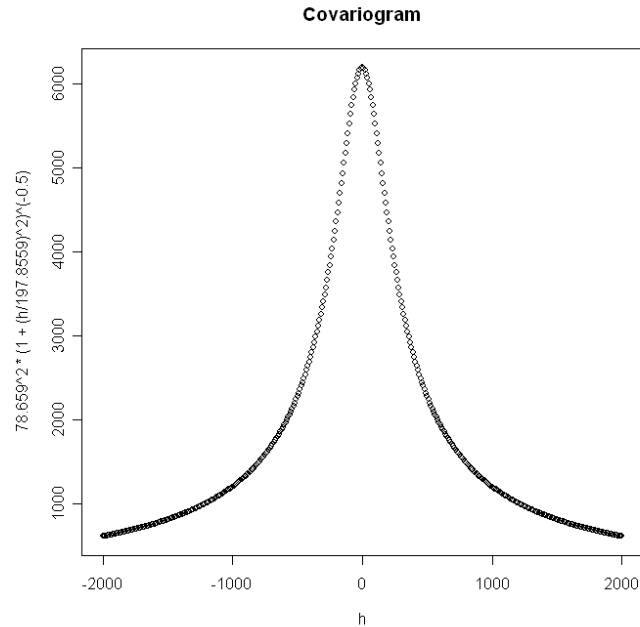
$$C(h) = 78.659^2 \cdot \left[1 + \left(\frac{h}{197.8559} \right)^2 \right]^{(-0.5)}$$

Checking for Validity

According to lecture note (p. 3), we should check for validity of covariogram model.

```
> h<-seq(-2000,2000,10)
```

```
> plot(h, 78.659^2 * (1 + (h/197.8559)^2)^(-0.5))
```



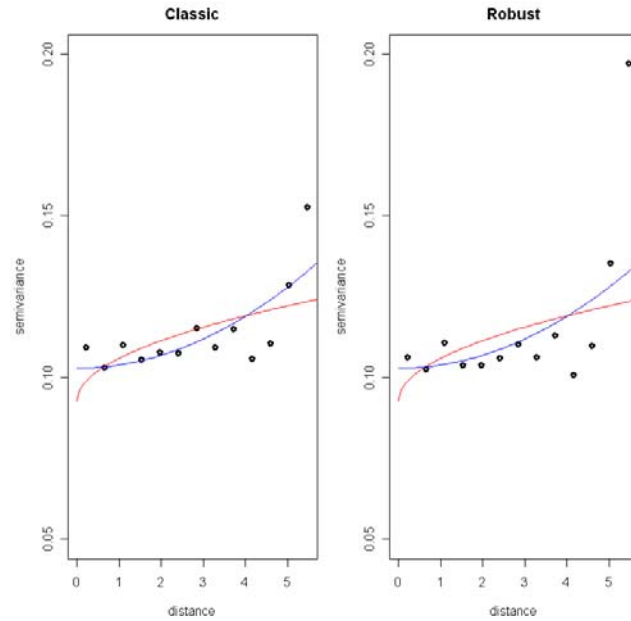
How can a proposed covariogram model be checked for validity? A sufficient condition in R is

- 1) $C(h) > 0$
- 2) $C(h) = C(-h)$
- 3) $C(h)$ is decreasing and convex in $h \in [0, \infty]$ (See Chung, 1968, Section 6.5)

In this section, empirical variograms (Classical and Robust) is fitted by two models: Cauchy and powered.exponential.

```
ini.vals <- expand.grid(seq(5,50,l=50), seq(5,50,l=50))
fit1a <- variofit(v1,ini=ini.vals,cov.model='powered.exponential',fix.nugget=FALSE,
  weights='equal',minimisation.function='optim')
fit1b <- variofit(v1,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
  weights='equal',minimisation.function='optim')
lines(fit1a,col=2)
lines(fit1b,col=4)

fit2a <- variofit(v2,ini=ini.vals,cov.model='powered.exponential',fix.nugget=FALSE,
  weights='equal',minimisation.function='optim')
fit2b <- variofit(v2,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
  weights='equal',minimisation.function='optim')
lines(fit1a,col=2)
lines(fit1b,col=4)
```



3. Spatial Prediction: Kriging

3.1 Without Demo. and Competition Data (Ordinary Kriging)

```
x <- seq(-86,-81,0.5)
y <- seq(30,35,0.5)
pts <- expand.grid(seq(-86,-81,0.5),seq(30,35,0.5))

#ini.vals <- expand.grid(seq(0,1,1=5), seq(0,1,1=5))

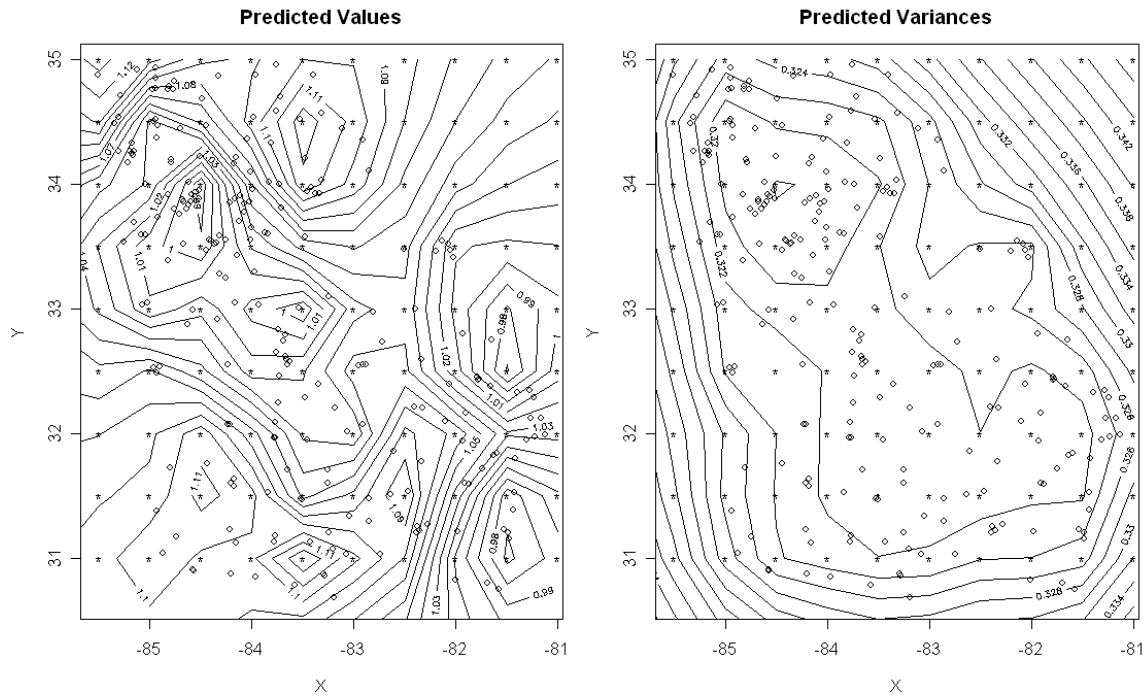
k <- krige.conv(coords=store.att$coords,data=store.att$data,
locations=pts,krige=krige.control(obj.m=fit1a))#ordinary Kriging

par(mfrow=c(1,2),mar=c(4,4,3,1))
p <- k$predict
z <- matrix(p,length(x),length(y))

plot(store.att$coords,main="Predicted Values")
#identify(store.att$coords,labels =store.att$data)
#identify(pts,labels =z,col=2)
points(pts,pch='*')
contour(x,y,z,nlevels=20,add=TRUE)
#filled.contour(x,y,z,nlevels=20)

p <- sqrt(k$krige.var)
z <- matrix(p,length(x),length(y))

plot(store.att$coords,main="Predicted Variances")
points(pts,pch='*')
contour(x,y,z,nlevels=20,add=TRUE)
```



A prediction location has large variance when it is far away from the observed sites.

Q: How good is my prediction? → Mean Squared Error of Prediction (MSPE):

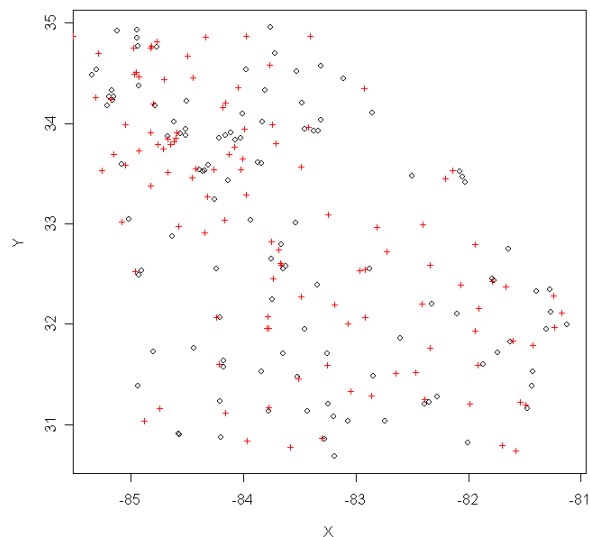
The data set is divided into two subsets:

Subset (a): Data values at sites numbered $j = 1, 3, 5, \dots, 225$. → Prediction

Subset (b): Data values at sites numbered $j = 2, 4, 6, \dots, 224$. → Modeling

Initially, subset (b) is the modeling set, and subset (a) is the prediction set.

```
> plot(subb.coords, xlab="X", ylab="Y") ##modeling
> points(suba.coords, col=2, pch='*') ##predicting
```

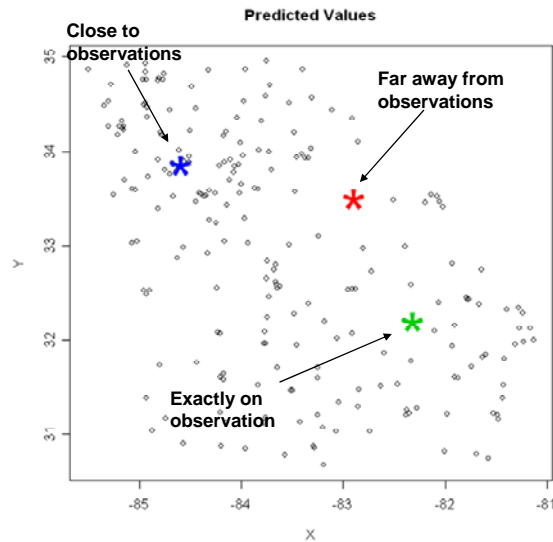


At above Figure, black circles indicate stores in subset (b). Red crosses are in subset (a). We predict the sales of stores in subset (a) by using the sales data of stores in subset (b), and then calculate MSPE for checking the accuracy of prediction.

$$MSPE = \frac{1}{113} \sum_{j=1,3,5...225} [Y(s_j) - \hat{Y}(s_j)]^2$$

```
v <- variog(coords=subb.coords,data=subb.data)
fit <- variofit(v,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
               weights='equal',minimisation.function='optim')
k <- krige.conv(coords=subb.coords,data=subb.data,
               locations=suba.coords,krige=krige.control(obj.m=fit))#ordinary Kriging
> MSPE=mean((p-suba.data)^2)
> MSPE
[1] 0.1053651
```

Q: What happens to MSPE if our prediction points are very close to given data, or very far from given data?



```
plot(store.att$coords,main="Predicted Values")
points(x=-82.9,y=33.5,col=2,pch='*',cex=5) # red
points(x=-84.6,y=33.85,col=4,pch='*',cex=5) # blue
points(x=-82.32230,y=32.20180,col=3,pch='*',cex=5) # green
aaa<-t(cbind(c(-82.9,33.5),c(-84.6,33.85),c(-82.32230,32.20180)))
k <- krige.conv(coords=store.att$coords,data=store.att$data,
               locations=aaa,krige=krige.control(obj.m=fitla))
```

```
> k
$predict
[1] 1.0603105 0.9699397 1.2189064

$krige.var
[1] 0.10561623 0.09974373 0.00000000
```

As prediction location far away from observed sites, prediction variance increases. Furthermore, we see no prediction variance at observed sites.

3.2 Variable Selection (stepwise regression)

We have demographics data as provided by Bureau of Census in 2000 and competition data. There are 27 variables related to sales.

```
> names(store)
[1] "Sales.2000"
[2] "Sales.Per.SF.2000"
[3] "DistanceToClosestCompetitorStore2000"
[4] "CompetitionIndex2000"
[5] "Population2000"
[6] "PCI2000"
[7] "PopulationGrowth2000"
[8] "NoOfHH2000"
[9] "HHGrowth2000"
[10] "AvgHHSIZE2000"
[11] "NumberOfFamilies2000"
[12] "FamilyGrowth2000"
[13] "PercentWhitePop2000"
[14] "PercentBlackPop2000"
[15] "PercentAsianPop2000"
[16] "PercentHispanicPop2000"
[17] "MedianAge2000"
[18] "PerCapitaIncome2000"
[19] "PercentHHInc0to252000"
[20] "PercentHHInc25to502000"
[21] "PercentHHInc50to1002000"
[22] "PercentHHInc100to1502000"
[23] "PercentHHIncAbove1502000"
[24] "MedianHHIncome2000"
[25] "Sq.Feet"
[26] "Longitude"
[27] "Latitude"
```

Since we have too many variables, we select 13 variables out of 27 by using stepwise algorithm. The effect of competitor data does not significant on the sales of stores. The selected variables explain sales data with 98% in R-squared.

```
> fit <- lm(Sales.2000/100000 ~ ., data=store)
> step(fit)
Step: AIC=-1354.93

Call:
lm(formula = Sales.2000/1e+05 ~ Sales.Per.SF.2000 + PCI2000 + PopulationGrowth2000 +
HHGrowth2000 + MedianAge2000 + PerCapitaIncome2000 + PercentHHInc0to252000 +
PercentHHInc25to502000 + PercentHHInc50to1002000 + PercentHHInc100to1502000 + Sq.Feet +
Longitude + Latitude, data = store)

Coefficients:
(Intercept)          Sales.Per.SF.2000          PCI2000
-2.809e+00           6.902e-02           3.131e-05
PopulationGrowth2000  HHGrowth2000          MedianAge2000
 1.729e-02          -1.269e-02          -2.679e-03
PerCapitaIncome2000  PercentHHInc0to252000  PercentHHInc25to502000
-2.460e-05           1.574e-02           1.509e-02
PercentHHInc50to1002000  PercentHHInc100to1502000  Sq.Feet
 1.406e-02           1.778e-02           1.491e-04
Longitude            Latitude
-5.686e-03          -6.737e-03

> summary(stepfit)
```

```
Call:
lm(formula = Sales.2000/1e+05 ~ Sales.Per.SF.2000 + PCI2000 +
    PopulationGrowth2000 + HHGrowth2000 + MedianAge2000 + PerCapitaIncome2000 +
    PercentHHInc0to252000 + PercentHHInc25to502000 + PercentHHInc50to1002000 +
    PercentHHInc100to1502000 + Sq.Feet + Longitude + Latitude,
    data = store)
```

```
Residuals:
    Min       1Q   Median       3Q      Max
-0.305322 -0.021674  0.003615  0.024713  0.165851
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      -2.809e+00  6.331e-01  -4.438 1.46e-05 ***
Sales.Per.SF.2000  6.902e-02  7.253e-04  95.158 < 2e-16 ***
PCI2000           3.131e-05  1.524e-05   2.054  0.04119 *
PopulationGrowth2000 1.729e-02  7.719e-03   2.241  0.02610 *
HHGrowth2000     -1.269e-02  5.683e-03  -2.233  0.02660 *
MedianAge2000    -2.679e-03  1.371e-03  -1.954  0.05198 .
PerCapitaIncome2000 -2.460e-05  1.517e-05  -1.622  0.10630
PercentHHInc0to252000 1.574e-02  5.754e-03   2.736  0.00674 **
PercentHHInc25to502000 1.509e-02  5.427e-03   2.780  0.00593 **
PercentHHInc50to1002000 1.406e-02  5.123e-03   2.745  0.00658 **
PercentHHInc100to1502000 1.778e-02  7.647e-03   2.325  0.02105 *
Sq.Feet          1.491e-04  2.854e-06  52.257 < 2e-16 ***
Longitude        -5.686e-03  3.435e-03  -1.655  0.09932 .
Latitude         -6.737e-03  3.221e-03  -2.092  0.03766 *
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.04778 on 211 degrees of freedom
Multiple R-Squared:  0.9802,    Adjusted R-squared:  0.979
F-statistic: 804.2 on 13 and 211 DF,  p-value: < 2.2e-16
```

3.3 With Demo. and Competition Data

The implicit model assumes that there is an underlying process with mean $\mu(x)$, where $x = (x_1, x_2)$ denotes the coordinates of a spatial location. The argument `trend` defines the form of the mean and the following options are allowed:

Ordinary Kriging

"cte"

the mean is assumed to be constant over the region, in which case $\mu(x) = \mu$.

This is the default option.

```
> k0 <- krige.conv(coords=subb.coords,data=subb.data,
+ locations=suba.coords,krige=krige.control(obj.m=fit,
+ trend.d = trend.spatial("cte", geodata=as.geodata(subb.demo,coords.col = 1:2,data.col =
+ 3,covar.col=4:14)),
+ trend.l = trend.spatial("cte", geodata=as.geodata(suba.demo,coords.col =
+ 1:2,data.col = 3,covar.col=4:14))
+ )
+ )
krige.conv: model with covariates matrix provided by the user
krige.conv: Kriging performed using global neighbourhood
> mean((k0$predict-suba.data)^2)#[1] 0.1053651
```

```
[1] 0.1053651
```

Universal Kriging

"1st"

the mean is assumed to be a first order polynomial on the coordinates:

$$\mu(x) = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2.$$

```
#ordinary Kriging
k1 <- krige.conv(coords=subb.coords,data=subb.data,
locations=suba.coords,krige=krige.control(obj.m=fit,
trend.d = trend.spatial("1st", geodata=as.geodata(subb.demo,coords.col =
1:2,data.col = 3,covar.col=4:14)),
trend.l = trend.spatial("1st", geodata=as.geodata(suba.demo,coords.col =
1:2,data.col = 3,covar.col=4:14))
)
)
> mean((k1$predict-suba.data)^2)#0.00395828
[1] 0.1062101
```

"2nd"

the mean is assumed to be a second order polynomial on the coordinates:

$$\mu(x) = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * (x_1)^2 + \beta_4 * (x_2)^2 + \beta_5 * x_1 * x_2.$$

```
> mean((k1$predict-suba.data)^2)#0.1062101
[1] 0.1068561
```

Universal Kriging with external trend: using demographic information

~ model

a model specification. See `formula` for further details on how to specify a model in R using formulas. Notice that the model term before the ~ is not necessary.

Typically used to include covariates (external trend) in the model.

```
k3 <- krige.conv(coords=subb.coords,data=subb.data,
locations=suba.coords,krige=krige.control(obj.m=fit,
trend.d = trend.spatial("2nd", geodata=as.geodata(subb.demo,coords.col =
1:2,data.col = 3,covar.col=4:14),
add =
~covar1+covar2+covar3+covar4+covar5+covar6+covar7+covar8+covar9+covar10+covar11),
trend.l = trend.spatial("2nd", geodata=as.geodata(suba.demo,coords.col =
1:2,data.col = 3,covar.col=4:14),
add =
~covar1+covar2+covar3+covar4+covar5+covar6+covar7+covar8+covar9+covar10+covar11)
)
)
> mean((k3$predict-suba.data)^2)#
[1] 0.004043961
```

Therefore, demographic information improves the accuracy of Kriging prediction.

4. Appendix

```
##### install packages
library(sp)
library(geoR)
library(scatterplot3d)

##### 1. read data
store<-read.csv("store_data_2000.csv")

data<-(store$Sales.2000)/100000
X<-store$Longitude
Y<-store$Latitude
store.att<-cbind(X,Y,data)
store.att<- as.geodata(store.att)

store.demo<-cbind(X,Y,data,store$Sales.Per.SF.2000,store$PCI2000,store$PopulationGrowth2000,
store$HHGrowth2000,store$MedianAge2000,store$PerCapitaIncome2000,
store$PercentHHInc0to252000 ,store$PercentHHInc25to502000, store$PercentHHInc50to1002000,
store$PercentHHInc100to1502000,store$Sq.Feet)

##### plotting of raw data
plot(store.att)

for (i in 1:3600)
scatterplot3d(store.att$coords[,1],store.att$coords[,2],store.att$data,
types="h",xlab="X", ylab="Y", zlab="Sales", angle=i*0.5,main="Store sales over given region")

##### 2. Drawing Variogram for Classic
par(mfrow=c(1,3),mar=c(4,4,3,1))
ini.vals <- expand.grid(seq(0,1,l=5), seq(0,1,l=5))

v1 <- variog(coords=store.att$coords,data=store.att$data)
plot(v1,main="Classic",lwd=3,ylim=c(0.05,0.2))
fit1a <- variofit(v1,ini=ini.vals,cov.model='powered.exponential',fix.nugget=FALSE,
weights='equal',minimisation.function='optim')
lines(fit1a,col=2)
fit1b <- variofit(v1,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
weights='equal',minimisation.function='optim')
lines(fit1b,col=4)
##### Drawing Variogram for Robust
v2<-variog(coords=store.att$coords,data=store.att$data,estimator.type = c("modulus"))
plot(v2,main="Robust",lwd=3,ylim=c(0.05,0.2))
fit2a <- variofit(v2,ini=ini.vals,cov.model='powered.exponential',fix.nugget=FALSE,
weights='equal',minimisation.function='optim')
lines(fit1a,col=2)

fit2b <- variofit(v2,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
weights='equal',minimisation.function='optim')
lines(fit1b,col=4)

#####compare Classic vs. Robust
plot(v1$v,v2$v,lwd=3,main="Classic vs. Robust")

##### kriging on grid
x <- seq(-86,-81,0.5)
y <- seq(30,35,0.5)
pts <- expand.grid(seq(-86,-81,0.5),seq(30,35,0.5)) # grid

#ini.vals <- expand.grid(seq(0,1,l=5), seq(0,1,l=5))

k <- krige.conv(coords=store.att$coords,data=store.att$data,
locations=pts,krige=krige.control(obj.m=fit1a)) #ordinary Kriging

par(mfrow=c(1,2),mar=c(4,4,3,1))
p <- k$predict
z <- matrix(p,length(x),length(y))

plot(store.att$coords,main="Predicted Values")
#identify(store.att$coords,labels =store.att$data)
#identify(pts,labels =z,col=2)
points(pts,pch='*')
contour(x,y,z,nlevels=20,add=TRUE)
#filled.contour(x,y,z,nlevels=20)

p <- sqrt(k$krige.var)
z <- matrix(p,length(x),length(y))
```

```

plot(store.att$coords,main="Predicted Variances")
points(pts,pch='*')
contour(x,y,z,nlevels=20,add=TRUE)

##### Divided into Two subsets
suba.coords<-matrix(0,113,2)
subb.coords<-matrix(0,112,2)
suba.data<-numeric(113)
subb.data<-numeric(112)
suba.demo<-matrix(0,113,14)
subb.demo<-matrix(0,112,14)

for (i in 1:225){
  if(i%%2==1){
    suba.coords[(i+1)/2,]<-store.att$coords[i,]
    suba.data[(i+1)/2]<-store.att$data[i]
    suba.demo[(i+1)/2,]<-store.demo[i,]
  }
  else{
    subb.coords[i/2,]<-store.att$coords[i,]
    subb.data[i/2]<-store.att$data[i]
    subb.demo[i/2,]<-store.demo[i,]
  }
}
par(mfrow=c(1,1),mar=c(4,4,3,1))
plot(subb.coords,xlab="X",ylab="Y") ##modeling
points(suba.coords,col=2,pch='+') ##predicting

##### compare prediction variance in terms of prediction locations

plot(store.att$coords,main="Predicted Values")
points(x=-82.9,y=33.5,col=2,pch='*',cex=5)
points(x=-84.6,y=33.85,col=4,pch='*',cex=5)
points(x=-82.32230,y=32.20180,col=3,pch='*',cex=5)
aaa<-t(cbind(c(-82.9,33.5),c(-84.6,33.85),c(-82.32230, 32.20180)))
k <- krige.conv(coords=store.att$coords,data=store.att$data,
locations=aaa,krige=krige.control(obj.m=fitla))
k

#####model fitting with modeling data(subb.coord, subb.data)

ini.vals <- expand.grid(seq(0,1,l=5), seq(0,1,l=5))
v <- variog(coords=subb.coords,data=subb.data)
fit <- variofit(v,ini=ini.vals,cov.model='cauchy',fix.nugget=FALSE,
weights='equal',minimisation.function='optim')

##### ordinary Kriging
k0 <- krige.conv(coords=subb.coords,data=subb.data,
locations=suba.coords,krige=krige.control(obj.m=fit,
trend.d = trend.spatial("cte", geodata=as.geodata(subb.demo,coords.col = 1:2,data.col =
3,covar.col=4:14)),
trend.l = trend.spatial("cte", geodata=as.geodata(suba.demo,coords.col = 1:2,data.col =
3,covar.col=4:14))
)
)
mean((k0$predict-suba.data)^2)#[1] 0.1053651

##### universal Kriging with a second order polynomial on the coordinates
##### you can choose polynomial order between (1st, 2nd)
k1 <- krige.conv(coords=subb.coords,data=subb.data,
locations=suba.coords,krige=krige.control(obj.m=fit,
trend.d = trend.spatial("2nd", geodata=as.geodata(subb.demo,coords.col = 1:2,data.col =
3,covar.col=4:14)),
trend.l = trend.spatial("2nd", geodata=as.geodata(suba.demo,coords.col = 1:2,data.col =
3,covar.col=4:14))
)
)
mean((k1$predict-suba.data)^2)

##### Universal Kriging with external trend
k3 <- krige.conv(coords=subb.coords,data=subb.data,
locations=suba.coords,krige=krige.control(obj.m=fit,
trend.d = trend.spatial("2nd", geodata=as.geodata(subb.demo,coords.col = 1:2,data.col =
3,covar.col=4:14),add =
~covar1+covar2+covar3+covar4+covar5+covar6+covar7+covar8+covar9+covar10+covar11),
trend.l = trend.spatial("2nd", geodata=as.geodata(suba.demo,coords.col = 1:2,data.col =
3,covar.col=4:14),add =
~covar1+covar2+covar3+covar4+covar5+covar6+covar7+covar8+covar9+covar10+covar11)
)
)
mean((k3$predict-suba.data)^2)

```