Supplementary Material of the paper

New robust cross-variogram estimators and approximations of their distributions based on saddlepoint techniques

Approximation VOM+SAD (10) for the Method-of-Moments estimator

If `rhoi`= $\sqrt{2\gamma_{ii}(\mathbf{h})}$, `rhoj`= $\sqrt{2\gamma_{jj}(\mathbf{h})}$, `gammaij`= $2\gamma_{ij}(\mathbf{h})$

```
napprox<-function(n,rhoi,rhoj,gammaij,t,epsilon,g)
{
rho<-gammaij/(rhoi*rhoj)
a<-((rhoi*rhoj)^2)*t*(1-rho^2)
b<-((rhoi*rhoj)^2)-(gammaij^2)+2*(rhoi*rhoj)*t
c<-gammaij-t
z0<-(-b+sqrt(b^2-4*a*c) )/(2*a)
K<--z0*t-0.5*log(1-z0*rhoi*rhoj*(1+rho))-0.5*log(1+z0*rhoi*rhoj*(1-rho))
s<-sqrt(-2*n*K)
K2<-0.5*((rhoi*rhoj*(1+rho))^2)/ ((1-z0*rhoi*rhoj*(1+rho))^2)+0.5*((rhoi*rhoj*(1-rho))^2)/
    ((1-z0*rhoi*rhoj*(1-rho))^2)
r1<-z0*sqrt(K2)
a1<-n/(1-rho)
a2<-n/(1+rho)
   integrando<-function(z){
   (n^(0.5*(n+1)))  *  (2^(0.5*(1-n)))*  ((abs(z))^(0.5*(n-1))) *
   ( 1/(gamma(n/2)*sqrt(pi*(1-rho^2))))*
   exp(0.5*(a1-a2)*z)*besselK(0.5*(a1+a2)*abs(z),0.5*(1-n))
   }
A<-integrate(integrando,t/(rhoi*rhoj),4)$value
B<-(exp(-z0*t))*((1-z0*(g^2)*rhoi*rhoj*(1+rho))^(-0.5))*((1+z0*(g^2)*rhoi*rhoj*(1-rho))^(-0.5))
C<-(exp(-z0*t))*((1-z0*rhoi*rhoj*(1+rho))^(-0.5))*((1+z0*rhoi*rhoj*(1-rho))^(-0.5))
result<-A+epsilon*sqrt(n)*dnorm(s)/r1*(B/C-1)
return(result)
}
```

Remark 2

If $N_h = n$ is large, we use

```
napprox2<-function(n,rhoi,rhoj,gammaij,t,epsilon,g)
{
rho<-gammaij/(rhoi*rhoj)
a<-((rhoi*rhoj)^2)*t*(1-rho^2)
b<-((rhoi*rhoj)^2)-(gammaij^2)+2*(rhoi*rhoj)*t
c<-gammaij-t
z0<-(-b+sqrt(b^2-4*a*c) )/(2*a)
K<--z0*t-0.5*log(1-z0*rhoi*rhoj*(1+rho))-0.5*log(1+z0*rhoi*rhoj*(1-rho))
s<-sqrt(-2*n*K)
K2<-0.5*((rhoi*rhoj*(1+rho))^2)/ ((1-z0*rhoi*rhoj*(1+rho))^2)+0.5*((rhoi*rhoj*(1-rho))^2)/
        ((1-z0*rhoi*rhoj*(1-rho))^2)
r1<-z0*sqrt(K2)
a1<-n/(1-rho)
a2<-n/(1+rho)
Lea<-1-pnorm(((t/(rhoi*rhoj)-rho)*sqrt(n))/(sqrt(1+rho^2)))
B<-(exp(-z0*t))*((1-z0*(g^2)*rhoi*rhoj*(1+rho))^(-0.5))*((1+z0*(g^2)*rhoi*rhoj*(1-rho))^(-0.5))
C<-(exp(-z0*t))*((1-z0*rhoi*rhoj*(1+rho))^(-0.5))*((1+z0*rhoi*rhoj*(1-rho))^(-0.5))
result<-Lea+epsilon*sqrt(n)*dnorm(s)/r1*(B/C-1)
return(result)
}
```

Figure 1 of the paper

4.1 Performance of the theoretical results with simulations.

We define a simulation function

```
library(MASS)

simula1<-
function(n,B,e,rhoi,rhoj,gammaij,g,t)
{
s<-NULL
for(j in 1:B){
M<-mvrnorm(n,mu=c(0,0),Sigma=matrix(c(rhoi^2,gammaij,gammaij,rhoj^2),ncol=2))
s[j]<-mean(
(sqrt( (1-e)*rhoi^2*(M[,1]/rhoi)^2+e*g^2*rhoi^2*(M[,1]/rhoi)^2 )   )*
(sqrt( (1-e)*rhoj^2*(M[,2]/rhoj)^2+e*g^2*rhoj^2*(M[,2]/rhoj)^2  )   )
)
}
P=ecdf(s)
1-P(t)
}
```

and we check the theoretical results we obtain by applying `napprox` function, considering several situations of contamination. In all of them we have assumed that it is `rhoi`$= 0.5$, `rhoj`$= 0.7$, `gammaij`$= 0.3$ and the sample size is `n`$= 3$.

No Contamination, $\boldsymbol{\epsilon = 0}$

```
library(MASS)
par(mfrow=c(2,2))
dibujo<-function(B)
{
t<-seq(0,B,len=100)
s<-NULL
for(j in 1:100){
s[j]<-napprox(3,0.5,0.7,0.3,t[j],0,1.1)
}
points(t,s,type="l",lwd=2)
}

t<-seq(0,1,len=100)
plot(t,simula1(3,100000,0,0.5,0.7,0.3,1.1,t),type="l",col=2,lwd=2,ylab=" ",
```

```
main="No contamination. epsilon=0")

dibujo(1)
```

Contamination $\epsilon = \mathbf{0.05}$

```
dibujo<-function(B)
{
t<-seq(0,B,len=100)
s<-NULL
for(j in 1:100){
s[j]<-napprox(3,0.5,0.7,0.3,t[j],0.05,1.1)
}
points(t,s,type="l",lwd=2)
}

t<-seq(0,1,len=100)
plot(t,simula1(3,100000,0.05,0.5,0.7,0.3,1.1,t),type="l",col=2,lwd=2,ylab=" ",
main="epsilon=0.05")

dibujo(1)
```

Contamination $\epsilon = \mathbf{0.1}$

```
dibujo<-function(B)
{
t<-seq(0,B,len=100)
s<-NULL
for(j in 1:100){
s[j]<-napprox(3,0.5,0.7,0.3,t[j],0.1,1.1)
}
points(t,s,type="l",lwd=2)
}

t<-seq(0,1,len=100)
plot(t,simula1(3,100000,0.1,0.5,0.7,0.3,1.1,t),type="l",col=2,lwd=2,ylab=" ",
main="epsilon=0.1")

dibujo(1)
```

Contamination $\boldsymbol{\epsilon = 0.2}$

```
dibujo<-function(B)
{
t<-seq(0,B,len=100)
s<-NULL
for(j in 1:100){
s[j]<-napprox(3,0.5,0.7,0.3,t[j],0.2,1.1)
}
points(t,s,type="l",lwd=2)
}

t<-seq(0,1,len=100)
plot(t,simula1(3,100000,0.2,0.5,0.7,0.3,1.1,t),type="l",col=2,lwd=2,ylab=" ",
main="epsilon=0.2")

dibujo(1)
```

Table 1:

```
NO CONTAMINATION, epsilon=0

t=0.4
> napprox(3,0.5,0.7,0.3, 0.4 ,0,1.1)
[1] 0.2662735
> simula1(3,100000,0,0.5,0.7,0.3,1.1,0.4)
[1] 0.27903

t=0.6
> napprox(3,0.5,0.7,0.3, 0.6 ,0,1.1)
[1] 0.119441
> simula1(3,100000,0,0.5,0.7,0.3,1.1,0.6)
[1] 0.12512

t=0.8
> napprox(3,0.5,0.7,0.3, 0.8 ,0,1.1)
[1] 0.05051644
> simula1(3,100000,0,0.5,0.7,0.3,1.1,0.8)
[1] 0.05665

t=0.9
> napprox(3,0.5,0.7,0.3, 0.9 ,0,1.1)
[1] 0.0318918
> simula1(3,100000,0,0.5,0.7,0.3,1.1,0.9)
[1] 0.03671

t=1
> napprox(3,0.5,0.7,0.3, 1 ,0,1.1)
[1] 0.01949519
> simula1(3,100000,0,0.5,0.7,0.3,1.1,1)
[1] 0.02451

-------------------------------------
epsilon=0.05


t=0.4
> napprox(3,0.5,0.7,0.3, 0.4 ,0.05,1.1)
[1] 0.2710139
> simula1(3,100000,0.05,0.5,0.7,0.3,1.1,0.4)
[1] 0.28199

t=0.6
> napprox(3,0.5,0.7,0.3, 0.6 ,0.05,1.1)
[1] 0.1231876
> simula1(3,100000,0.05,0.5,0.7,0.3,1.1,0.6)
[1] 0.12904
```

```
t=0.8
> napprox(3,0.5,0.7,0.3, 0.8 ,0.05,1.1)
[1] 0.05301946
> simula1(3,100000,0.05,0.5,0.7,0.3,1.1,0.8)
[1] 0.05895

t=0.9
> napprox(3,0.5,0.7,0.3, 0.9 ,0.05,1.1)
[1] 0.03386354
> simula1(3,100000,0.05,0.5,0.7,0.3,1.1,0.9)
[1] 0.03727

t=1
> napprox(3,0.5,0.7,0.3, 1 ,0.05,1.1)
[1] 0.0210216
> simula1(3,100000,0.05,0.5,0.7,0.3,1.1,1)
[1] 0.02449

------------------------------------
epsilon=0.2

t=0.4
> napprox(3,0.5,0.7,0.3, 0.4 ,0.2,1.1)
[1] 0.285235
> simula1(3,100000,0.2,0.5,0.7,0.3,1.1,0.4)
[1] 0.29635

t=0.6
> napprox(3,0.5,0.7,0.3, 0.6 ,0.2,1.1)
[1] 0.1344277
> simula1(3,100000,0.2,0.5,0.7,0.3,1.1,0.6)
[1] 0.13795

t=0.8
> napprox(3,0.5,0.7,0.3, 0.8 ,0.2,1.1)
[1] 0.06052852
> simula1(3,100000,0.2,0.5,0.7,0.3,1.1,0.8)
[1] 0.06312

t=0.9
> napprox(3,0.5,0.7,0.3, 0.9 ,0.2,1.1)
[1] 0.03977877
> simula1(3,100000,0.2,0.5,0.7,0.3,1.1,0.9)
[1] 0.04304

t=1
> napprox(3,0.5,0.7,0.3, 1 ,0.2,1.1)
[1] 0.02560081
> simula1(3,100000,0.2,0.5,0.7,0.3,1.1,1)
```

```
[1] 0.02811
```

The values of the VOM + SAD approximation are slightly lower because the VOM approximation is a first order approximation. However, the SAD approach is second-order.

Figure 2 of the paper

We assume that it is `rhoi`= 1.5, `rhoj`= 1.8, `gammaij`= 2.

```
dibu2<-function(B)
{
t<-seq(4,B,len=100)
s<-NULL
w1<-NULL
w2<-NULL
for(j in 1:100){
s[j]<-napprox(3,1.5,1.8,2,t[j],0,1.1)
w1[j]<-napprox(3,1.5,1.8,2,t[j],0.15,1.1)
w2[j]<-napprox(3,1.5,1.8,2,t[j],0.3,1.1)
}
plot(t,s,type="l",main="(1-epsilon)*N(0,1)+epsilon*N(0,1.21)",ylab=" ",lwd=2)
lines(t,w1,type="l",col=2,lty=2,lwd=2)
lines(t,w2,type="l",col=4,lty=4,lwd=2)
legend(7,0.12,c("epsilon=0","epsilon=0.15","epsilon=0.3"),lty=c(1,2,4),col=c(1,2,4),
  cex=0.8,xjust=0)
}


dibu3<-function(B)
{
t<-seq(4,B,len=100)
s<-NULL
w1<-NULL
w2<-NULL
for(j in 1:100){
s[j]<-napprox(3,1.5,1.8,2,t[j],0,1.2)
w1[j]<-napprox(3,1.5,1.8,2,t[j],0.15,1.2)
w2[j]<-napprox(3,1.5,1.8,2,t[j],0.3,1.2)}
plot(t,s,type="l",main="(1-epsilon)*N(0,1)+epsilon*N(0,1.44)",ylab=" ",lwd=2)
lines(t,w1,type="l",col=2,lty=2,lwd=2)
lines(t,w2,type="l",col=4,lty=4,lwd=2)
legend(7,0.12,c("epsilon=0","epsilon=0.15","epsilon=0.3"),lty=c(1,2,4),col=c(1,2,4),
  cex=0.8,xjust=0)
}


par(mfrow=c(1,2))
dibu2(14)
dibu3(14)
```

Figure 3 of the paper

We assume that it is rhoi= 1.5, rhoj= 1.8, gammaij= 2.

```
dibu12<-function(B,n)
{
t<-seq(4,B,len=100)
s<-NULL
w1<-NULL
w2<-NULL
for(j in 1:100){
s[j]<-napprox(n,1.5,1.8,2,t[j],0,1.1)
w1[j]<-napprox(n,1.5,1.8,2,t[j],0.15,1.1)
w2[j]<-napprox(n,1.5,1.8,2,t[j],0.3,1.1)
}
plot(t,s,type="l",main="(1-epsilon)*N(0,1)+epsilon*N(0,1.21)",ylab=" ",lwd=2, sub="n=10"
     ,ylim=c(0,0.05))
lines(t,w1,type="l",col=2,lty=2,lwd=2)
lines(t,w2,type="l",col=4,lty=4,lwd=2)
legend(4.75,0.05,c("epsilon=0","epsilon=0.15","epsilon=0.3"),lty=c(1,2,4),col=c(1,2,4)
     ,cex=0.8,xjust=0)
}
```

```
dibu13<-function(B,n)
{
t<-seq(4,B,len=100)
s<-NULL
w1<-NULL
w2<-NULL
for(j in 1:100){
s[j]<-napprox(n,1.5,1.8,2,t[j],0,1.2)
w1[j]<-napprox(n,1.5,1.8,2,t[j],0.15,1.2)
w2[j]<-napprox(n,1.5,1.8,2,t[j],0.3,1.2)}
plot(t,s,type="l",main="(1-epsilon)*N(0,1)+epsilon*N(0,1.44)",ylab=" ",lwd=2, sub="n=10"
     ,ylim=c(0,0.05))
lines(t,w1,type="l",col=2,lty=2,lwd=2)
lines(t,w2,type="l",col=4,lty=4,lwd=2)
}
```

```
dibu21<-function(B,alpha,ite,n)
{
t<-seq(4,B,len=100)
c2<-(1/((1-2*alpha)^(1/(ite+1))))-1
c1<-((1-2*alpha)^(1/(ite+1)))-1
s<-NULL
w1<-NULL
w2<-NULL
```

```
for(j in 1:100){
s[j]<-((1+n*c1)^(ite+1))*((1+n*c2)^(ite+1))*napprox(n,1.5,1.8,2,t[j],0,1.1)
w1[j]<-((1+n*c1)^(ite+1))*((1+n*c2)^(ite+1))*napprox(n,1.5,1.8,2,t[j],0.15,1.1)
w2[j]<-((1+n*c1)^(ite+1))*((1+n*c2)^(ite+1))*napprox(n,1.5,1.8,2,t[j],0.3,1.1)
}
plot(t,s,type="l",main="(1-epsilon)*N(0,1)+epsilon*N(0,1.21)",ylab=" ",lwd=2
     , sub="n=10.   0.2-Trimmed",ylim=c(0,0.05))
lines(t,w1,type="l",col=2,lty=2,lwd=2)
lines(t,w2,type="l",col=4,lty=4,lwd=2)
legend(4.75,0.05,c("epsilon=0","epsilon=0.15","epsilon=0.3"),lty=c(1,2,4),col=c(1,2,4)
     ,cex=0.8,xjust=0)
}


dibu22<-function(B,alpha,ite,n)
{
t<-seq(4,B,len=100)
c2<-(1/((1-2*alpha)^(1/(ite+1))))-1
c1<-((1-2*alpha)^(1/(ite+1)))-1
s<-NULL
w1<-NULL
w2<-NULL
for(j in 1:100){
s[j]<-((1+n*c1)^(ite+1))*((1+n*c2)^(ite+1))*napprox(n,1.5,1.8,2,t[j],0,1.2)
w1[j]<-((1+n*c1)^(ite+1))*((1+n*c2)^(ite+1))*napprox(n,1.5,1.8,2,t[j],0.15,1.2)
w2[j]<-((1+n*c1)^(ite+1))*((1+n*c2)^(ite+1))*napprox(n,1.5,1.8,2,t[j],0.3,1.2)}
plot(t,s,type="l",main="(1-epsilon)*N(0,1)+epsilon*N(0,1.44)",ylab=" ",lwd=2
     , sub="n=10.   0.2-Trimmed",ylim=c(0,0.05))
lines(t,w1,type="l",col=2,lty=2,lwd=2)
lines(t,w2,type="l",col=4,lty=4,lwd=2)
}


par(mfrow=c(2,2))
dibu12(5.5,10)
dibu13(5.5,10)
dibu21(5.5,0.2,100,10)
dibu22(5.5,0.2,100,10)
```

Example 1

(A) No contamination, $Z_1 \equiv N(0,1)$; (B) $Z_1 \equiv 0.95 \cdot N(0,1) + 0.05 \cdot N(0,5^2)$; (C) $Z_1 \equiv 0.90 \cdot N(0,1) + 0.10 \cdot N(0,5^2)$; (D) $Z_1 \equiv 0.80 \cdot N(0,1) + 0.20 \cdot N(0,5^2)$; (E) $Z_1 \equiv 0.95 \cdot N(0,1) + 0.05 \cdot N(0,20^2)$; (F) $Z_1 \equiv 0.90 \cdot N(0,1) + 0.10 \cdot N(0,20^2)$; (G) $Z_1 \equiv 0.80 \cdot N(0,1) + 0.20 \cdot N(0,20^2)$.

(A)

```
> Z1
 [1] -1.7067995  0.7593676 -1.2780228 -1.9793377 -0.1157498 -0.8611665
 [7] -1.5320051 -0.7397734  0.6411335  0.8620143  0.9618877  0.8179621
[13]  0.9728229 -0.1212191  0.3737695  2.1160674  1.1715363 -0.1973322
[19] -0.1993668  1.0457906
> Z2
 [1]  -7.7006664   4.5020855  -8.4446190 -10.5809381  -1.8419869  -3.1495547
 [7]  -8.7082110  -4.1244160   2.9707944   5.5003608   5.3501520   4.0005462
[13]   4.1519454  -0.1018844   1.7622045   7.9966436   6.8098809  -0.5119985
[19]  -1.4396256   5.3857130

> cor(Z1,Z2)
[1] 0.9834625
```

The coordinates we shall assume are

```
> Lon
 [1] 0.350 0.487 0.637 0.775 0.825 0.087 0.237 0.400 0.575 0.737 0.062 0.212
[13] 0.325 0.450 0.650 0.900 0.337 0.462 0.600 0.800
> Lat
 [1] 0.025 0.087 0.050 0.025 0.125 0.187 0.150 0.162 0.212 0.237 0.362 0.337
[13] 0.287 0.287 0.362 0.262 0.462 0.425 0.475 0.387
```

We have the data.frame with

```
> datasimula<-data.frame(Z1,Z2,Lon,Lat)
```

Because the coordinates in datasimula are `Lon` (longitude) and `Lat` (latitude), we establish these with the sentences

```
library(gstat)
library(sp)
coordinates(datasimula) = ~Lon+Lat
```

Then, we create a *gstat* object with

```
g <- gstat(NULL, "Z1", Z1 ~ 1, datasimula)
g <- gstat(g, "Z2", Z2 ~ 1, datasimula)
```

We compute, and plot, the variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations, with

```
vvm<-variogram(g,width=0.05)
plot(vvm)
```

To extract from `vvm` the values of the classical cross-variogram estimator we define

```
vvmcross<-vvm[vvm$id == "Z1.Z2",]
```

To compute the classical estimations (again) and the robust ones, we define

```
vm<-variogram(g,width=0.05,cloud=T)
vmcross<-vm[vm$id == "Z1.Z2",]
```

The previous classical estimations of the cross-variogram are obtained with

```
vvmcross$gamma
```

or with

```
mean(vmcross[vmcross$dist < 0.05,]$gamma)
mean(vmcross[vmcross$dist > 0.05 & vmcross$dist<0.10,]$gamma)
mean(vmcross[vmcross$dist > 0.10 & vmcross$dist<0.15,]$gamma)
mean(vmcross[vmcross$dist > 0.15 & vmcross$dist<0.20,]$gamma)
mean(vmcross[vmcross$dist > 0.20 & vmcross$dist<0.25,]$gamma)
mean(vmcross[vmcross$dist > 0.25 & vmcross$dist<0.30,]$gamma)
mean(vmcross[vmcross$dist > 0.30 & vmcross$dist<0.35,]$gamma)
mean(vmcross[vmcross$dist > 0.35 & vmcross$dist<0.40,]$gamma)
mean(vmcross[vmcross$dist > 0.40 & vmcross$dist<0.45,]$gamma)
mean(vmcross[vmcross$dist > 0.45,]$gamma)
```

The 0.1-trimmed variogram estimations are computed with

```
y1<-mean(vmcross[vmcross$dist < 0.05,]$gamma,0.1)
y2<-mean(vmcross[vmcross$dist > 0.05 & vmcross$dist<0.10,]$gamma,0.1)
y3<-mean(vmcross[vmcross$dist > 0.10 & vmcross$dist<0.15,]$gamma,0.1)
y4<-mean(vmcross[vmcross$dist > 0.15 & vmcross$dist<0.20,]$gamma,0.1)
y5<-mean(vmcross[vmcross$dist > 0.20 & vmcross$dist<0.25,]$gamma,0.1)
y6<-mean(vmcross[vmcross$dist > 0.25 & vmcross$dist<0.30,]$gamma,0.1)
y7<-mean(vmcross[vmcross$dist > 0.30 & vmcross$dist<0.35,]$gamma,0.1)
y8<-mean(vmcross[vmcross$dist > 0.35 & vmcross$dist<0.40,]$gamma,0.1)
y9<-mean(vmcross[vmcross$dist > 0.40 & vmcross$dist<0.45,]$gamma,0.1)
y10<-mean(vmcross[vmcross$dist > 0.45,]$gamma,0.1)

y<-c(y3,y4,y5,y6,y7)
```

The Huber's cross-variogram estimations are computed with

```
library(MASS)

u1<-huber(vmcross[vmcross$dist < 0.05,]$gamma,0.1)$mu
u2<-huber(vmcross[vmcross$dist > 0.05 & vmcross$dist<0.10,]$gamma)$mu
u3<-huber(vmcross[vmcross$dist > 0.10 & vmcross$dist<0.15,]$gamma)$mu
u4<-huber(vmcross[vmcross$dist > 0.15 & vmcross$dist<0.20,]$gamma)$mu
u5<-huber(vmcross[vmcross$dist > 0.20 & vmcross$dist<0.25,]$gamma)$mu
u6<-huber(vmcross[vmcross$dist > 0.25 & vmcross$dist<0.30,]$gamma)$mu
u7<-huber(vmcross[vmcross$dist > 0.30 & vmcross$dist<0.35,]$gamma)$mu
u8<-huber(vmcross[vmcross$dist > 0.35 & vmcross$dist<0.40,]$gamma)$mu
u9<-huber(vmcross[vmcross$dist > 0.40 & vmcross$dist<0.45,]$gamma)$mu
u10<-huber(vmcross[vmcross$dist > 0.45,]$gamma)$mu

u<-c(u3,u4,u5,u6,u7)
```

The classical, 0.1-trimmed and Huber's estimations are, respectively,

```
> vvmcross$gamma
[1] 2.747381 4.531952 5.624894 6.691009 5.113951
> y
[1] 2.526624 3.736630 4.168794 5.277753 5.093659
> u
[1] 2.417667 2.615297 3.146202 4.700480 5.113951
```

These estimations are very similar for each **h** and are plotted with

```
plot(vvmcross$dist,vvmcross$gamma,pch=16,ylab=" ",xlab="h",ylim=c(0,20))
points(vvmcross$dist,y,pch=16,col=3)
```

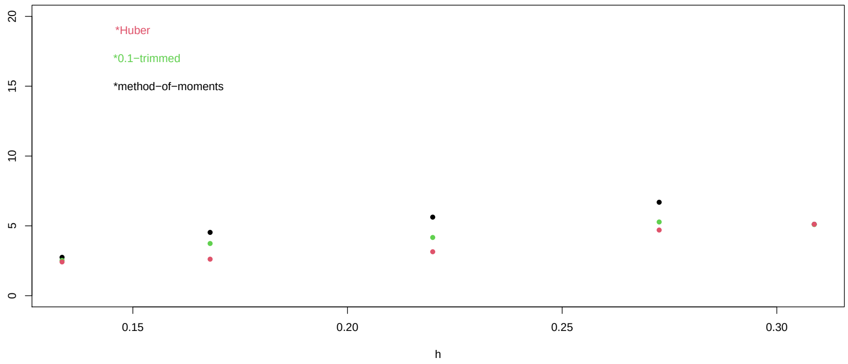

Figure 1: Variogram estimations with no contamination.

```
points(vvmcross$dist,u,pch=16,col=2)
text(0.15,15,"                       *method-of-moments")
text(0.15,17,"         *0.1-trimmed",col=3)
text(0.15,19,"*Huber",col=2)
```

obtaining Fig. 1.

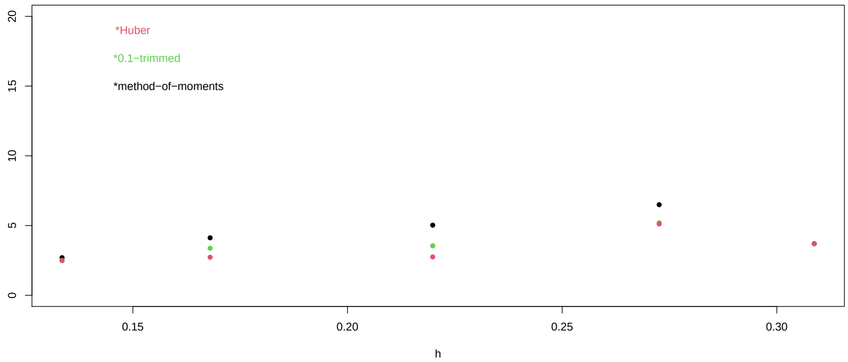

Figure 2: Variogram estimations with $Z1 \equiv 0.95 \cdot N(0,1) + 0.05 \cdot N(0,5^2)$.

(B)

The simulated observations were

```
> Z1
 [1] -1.54707637  0.72716315 -0.93864303 -1.92088422 -0.08832616 -0.81385872
 [7] -1.41786171 -0.58205613  0.82972893  0.89789483  0.97085287  0.74996793
[13]  1.20044704  0.03202161  0.27598535  2.10185314  0.92268707 -0.28437219
[19] -0.77709918  0.80217608
> Z2
 [1]  -7.24124986   4.28274518  -7.74690945 -10.09240460  -1.72825137
 [6]  -2.98782754  -8.23525736  -3.79746654   3.04290679   5.30432395
[11]   5.13970399   3.77342275   4.22061342   0.05038956   1.59499858
[16]   7.68840055   6.27911445  -0.58330517  -1.95534506   4.92510236
```

The classical estimations of the cross-variogram, the 0.1-trimmed variogram estimations and the Huber's cross-variogram estimations were

```
> vvmcross$gamma
[1] 2.695873 4.119727 5.027768 6.496513 3.700918
> y
[1] 2.498172 3.372803 3.548660 5.201775 3.690211
> u
[1] 2.484099 2.727849 2.749919 5.112608 3.700918
```

These estimations are plotted in Fig. 2.

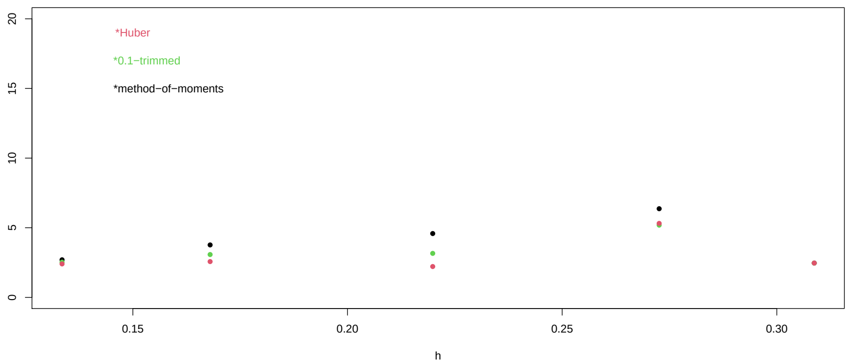

Figure 3: Variogram estimations with $Z1 \equiv 0.90 \cdot N(0,1) + 0.10 \cdot N(0,5^2)$.

(C)

The simulated observations were

```
> Z1
 [1] -1.38735321  0.69495870 -0.59926327 -1.86243071 -0.06090249 -0.76655092
 [7] -1.30371835 -0.42433882  1.01832435  0.93377532  0.97981808  0.68197373
[13]  1.42807114  0.18526234  0.17820120  2.08763893  0.67383786 -0.37141216
[19] -1.35483159  0.55856159
> Z2
 [1] -6.7818334  4.0634048 -7.0491999 -9.6038711 -1.6145158 -2.8261003
 [7] -7.7623037 -3.4705171  3.1150192  5.1082871  4.9292560  3.5462993
[13]  4.2892814  0.2026636  1.4277927  7.3801575  5.7483480 -0.6546118
[19] -2.4710645  4.4644917
```

The classical estimations of the cross-variogram, the 0.1-trimmed variogram estimations and the Huber's cross-variogram estimations were

```
> vvmcross$gamma
[1] 2.697672 3.765254 4.582922 6.364460 2.463898
> y
[1] 2.527690 3.075522 3.158054 5.186811 2.460909
> u
[1] 2.402753 2.570092 2.214797 5.311820 2.463898
```

These estimations are plotted in Fig. 3.

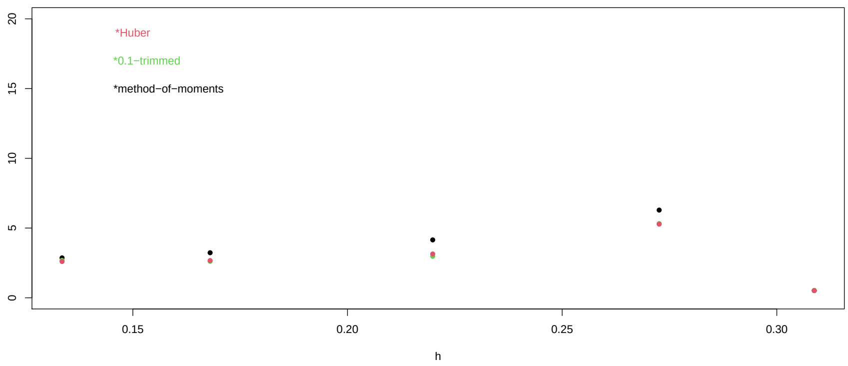

Figure 4: Variogram estimations with $Z1 \equiv 0.80 \cdot N(0,1) + 0.20 \cdot N(0,5^2)$.

(D)

The simulated observations were

```
> Z1
 [1] -1.067906890  0.630549783  0.079496235 -1.745523684 -0.006055142
 [6] -0.671935313 -1.075431627 -0.108904213  1.395515201  1.005536300
[11]  0.997748499  0.545985319  1.883319347  0.491743802 -0.017367116
[16]  2.059210516  0.176139456 -0.545492090 -2.510296418  0.071332605
> Z2
 [1] -5.8630004  3.6247241 -5.6537808 -8.6268040 -1.3870448 -2.5026459
 [7] -6.8163964 -2.8166182  3.2592439  4.7162135  4.5083600  3.0920525
[13]  4.4266173  0.5072115  1.0933809  6.7636715  4.6868152 -0.7972251
[19] -3.5025035  3.5432705
```

The classical estimations of the cross-variogram, the 0.1-trimmed variogram estimations and the Huber's cross-variogram estimations were

```
> vvmcross$gamma
[1] 2.8611951 3.2295654 4.1500696 6.2876804 0.5178936
> y
[1] 2.6804296 2.6175614 2.9758157 5.3169787 0.5294732
> u
[1] 2.6004649 2.6690677 3.1420181 5.2795846 0.5178936
```

These estimations are plotted in Fig. 4.

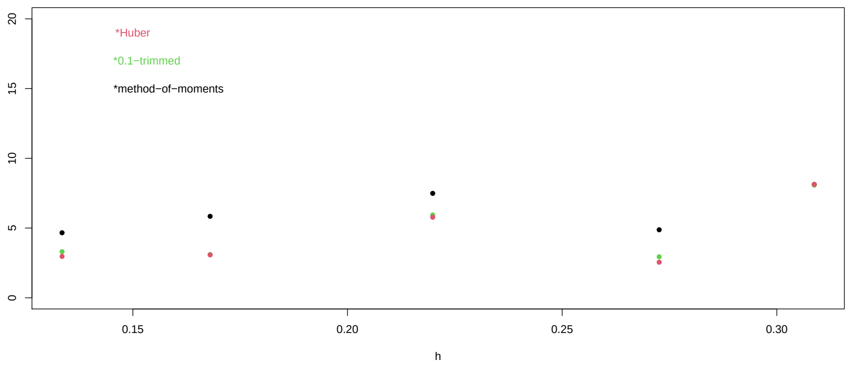

Figure 5: Variogram estimations with $Z1 \equiv 0.95 \cdot N(0,1) + 0.05 \cdot N(0,20^2)$.

(E)

The simulated observations were

```
> Z1
 [1] -1.1592920  2.8953017 -3.2508385 -1.7217333 -0.8665771 -1.0287402
 [7] -0.6991397 -1.7025192 -0.3346677  0.7867826 -0.6007092 -0.8419705
[13]  0.5359680 -0.4937611 -1.2399676  0.9013273  1.1684601 -1.5600846
[19]  1.1819173 -1.0655340
> Z2
 [1]  -6.853465532   6.450883724 -10.059104968  -9.893253658  -2.506502334
 [6]  -3.202709006  -7.516535386  -4.917929656   1.878510166   5.193211673
[11]   3.568141899   2.181484325   3.556134419  -0.475393195   0.079045598
[16]   6.487874736   6.524887520  -1.859017530   0.003671407   3.057392277
```

The classical estimations of the cross-variogram, the 0.1-trimmed variogram estimations and the Huber's cross-variogram estimations were

```
> vvmcross$gamma
[1] 4.662730 5.841062 7.486276 4.876525 8.129393
> y
[1] 3.305672 3.098810 5.937846 2.940312 8.078697
> u
[1] 2.965031 3.074587 5.775112 2.550679 8.129393
```

These estimations are plotted in Fig. 5.

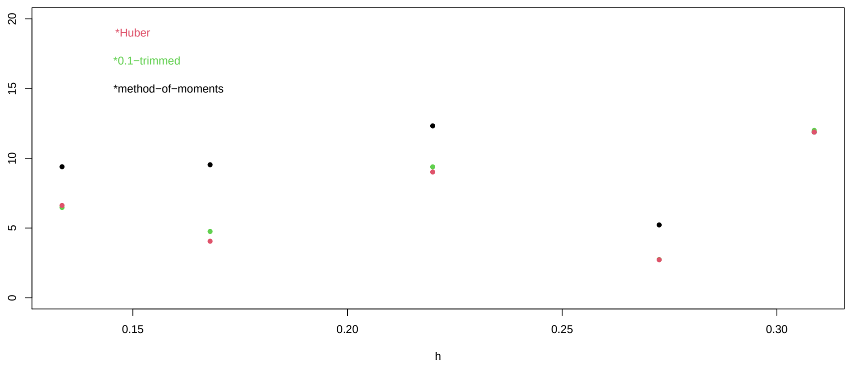

Figure 6: Variogram estimations with $Z1 \equiv 0.90 \cdot N(0,1) + 0.10 \cdot N(0,20^2)$.

(F)

The simulated observations were

```
> Z1
[1] -0.61178455  5.03123579 -5.22365431 -1.46412883 -1.61740441 -1.19631385
 [7]  0.13372559 -2.66526505 -1.31046889  0.71155077 -2.16330611 -2.50190312
[13]  0.09911315 -0.86630316 -2.85370477 -0.31341270  1.16538399 -2.92283688
[19]  2.56320135 -3.17685858
> Z2
 [1]  -6.0062647   8.3996819 -11.6735909  -9.2055692  -3.1710178  -3.2558633
 [7]  -6.3248598  -5.7114433   0.7862259   4.8860626   1.7861318   0.3624225
[13]   2.9603234  -0.8489020  -1.6041133   4.9791059   6.2398942  -3.2060365
[19]   1.4469684   0.7290716
```

The classical estimations of the cross-variogram, the 0.1-trimmed variogram estimations and the Huber's cross-variogram estimations were

```
> vvmcross$gamma
[1]  9.395201  9.538226 12.325172  5.223145 11.881194
> y
[1]  6.474346  4.756325  9.387484  2.741195 12.004038
> u
[1]  6.620501  4.054955  9.017632  2.726702 11.881194
```

These estimations are plotted in Fig. 6.

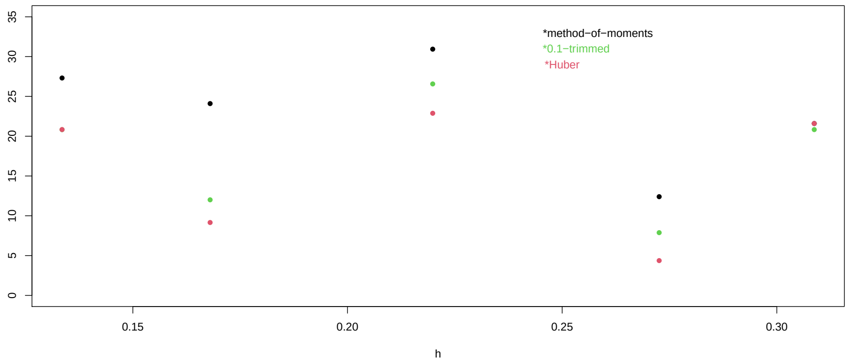

Figure 7: Variogram estimations with $Z1 \equiv 0.80 \cdot N(0,1) + 0.20 \cdot N(0,20^2)$.

(G)

The simulated observations were

```
> Z1
 [1]  0.4832304  9.3031040 -9.1692858 -0.9489199 -3.1190590 -1.5314612
 [7]  1.7994563 -4.5907567 -3.2620713  0.5610872 -5.2884999 -5.8217684
[13] -0.7745966 -1.6113872 -6.0811790 -2.7428928  1.1592317 -5.6483415
[19]  5.3257695 -7.3995077
> Z2
 [1]  -4.311863  12.297278 -14.902563  -7.830200  -4.500049  -3.362172
 [7]  -3.941508  -7.298471  -1.398343   4.271764  -1.777888  -3.275701
[13]   1.768701  -1.595919  -4.970431   1.961568   5.669907  -5.900075
[19]   4.333562  -3.927570
```

The classical estimations of the cross-variogram, the 0.1-trimmed variogram estimations and the Huber's cross-variogram estimations were

```
> vvmcross$gamma
[1] 27.31151 24.09672 30.93551 12.39970 21.59387
> y
[1] 20.820354 12.006113 26.566519  7.880707 20.826284
> u
[1] 20.833599  9.154754 22.879712  4.368788 21.593874
```

These estimations are plotted in Fig. 7.

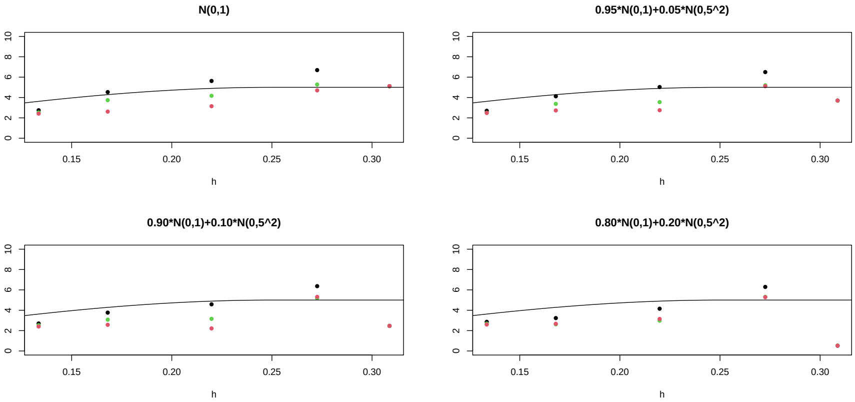

Figure 8: Variogram estimations of Example 1: classical (black), 0.1-trimmed (green) and Huber's (red), and the variogram model with no contamination.

We show in Fig. 8 and Fig. 9 several situations in which we observe the effect of the robust estimations when the outliers influence increases.

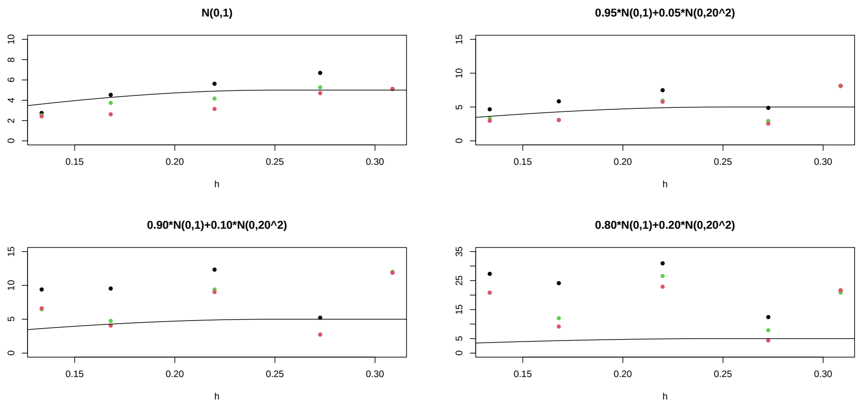

Figure 9: Variogram estimations of Example 1: classical (black), 0.1-trimmed (green) and Huber's (red), and the variogram model with no contamination.

Example 2

Let us consider *prediction data*, included in the jura data set, from Pierre Goovaerts' book, that contains geolocated information of several variables. This data set is called `prediction.dat` in the R library, `gstat`. This can be loaded with the sentence `data(jura)`.

Cross-variogram

Two correlated variables, with a distribution similar to a scale contaminated normal model, are `ln(Pb)` (natural logarithm of *lead*) and `Ni` (*nickel*). This can be observed with the following sentences:

```
library(gstat)
library(sp)
data(jura)
par(mfrow=c(1,2))
hist(log(prediction.dat$Pb),col=2)
hist(prediction.dat$Ni,col=2)
cor(log(prediction.dat$Pb),prediction.dat$Ni)
```

Because the coordinates in the jura data set are the variables `Xloc` (longitude) and `Yloc` (latitude), we use first the sentence

```
coordinates(prediction.dat) = ~Xloc+Yloc
```

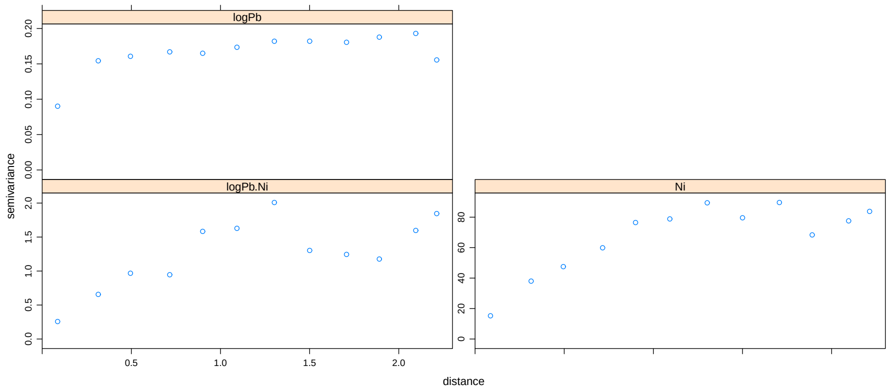

Figure 10: Variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations of Example 2.

Then we create a *gstat* object with

```
g<-gstat(NULL,"logPb",log(Pb)~1,prediction.dat)
g<-gstat(g,"Ni",Ni~1,prediction.dat)
```

We compute, and plot, the variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations, with

```
vvm<-variogram(g,width=0.2)
plot(vvm)
```

obtaining Fig. 10. From this cross-variogram plot we can admit, initially, an Spherical model with sill=psill=1.5, range=1 and nugget=0, model that is established with

```
model<-vgm(1.5,"Sph",1,0)
```

Now we check the adequacy between the model and the classical cross-variogram estimator with

```
> fit.lmc(vvm,g,model)

data:
logPb : formula = log(Pb)`~`1 ; data dim = 259 x 9
```

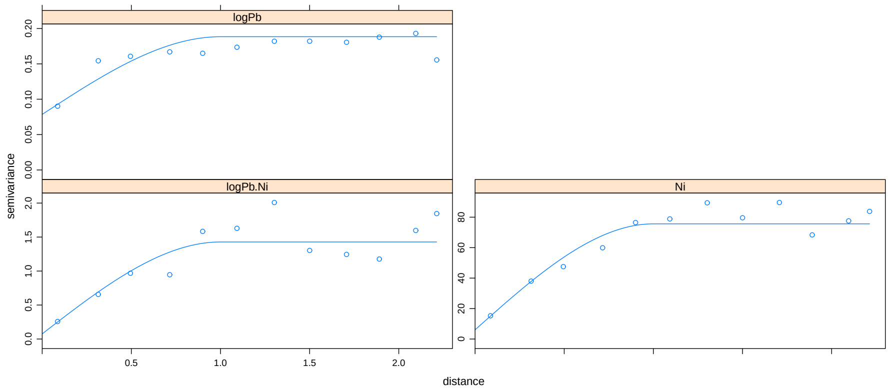

Figure 11: Variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations with the model of Example 2.

```
Ni : formula = Ni‘~‘1 ; data dim = 259 x 9
variograms:
            model       psill range
logPb[1]      Nug  0.07854047     0
logPb[2]      Sph  0.10983999     1
Ni[1]         Nug  6.03696691     0
Ni[2]         Sph 69.59106859     1
logPb.Ni[1]   Nug  0.07601742     0
logPb.Ni[2]   Sph  1.35202753     1
```

From the last two lines we conclude that a better model should be an Spherical model with partial sill=1.35202753, range=1 and nugget=0.07601742, model that we establish now

```
model2<-vgm(1.35202753,"Sph",1,0.07601742)
```

We obtain a picture of this, Fig. 11, with the sentence

```
plot(vvm,fit.lmc(vvm,g,model2))
```

To extract from `vvm` the values of the classical cross-variogram estimator, plotted in the previous picture, we define

```
vvmcross<-vvm[vvm$id == "logPb.Ni",]
```

To compute the classical estimations (again) and the robust ones, we define

```
vm<-variogram(g,width=0.2,cloud=T)
vmcross<-vm[vm$id == "logPb.Ni",]
```

The previous classical estimations of the cross-variogram are obtained with

```
vvmcross$gamma
```

or with

```
mean(vmcross[vmcross$dist < 0.2,]$gamma)
mean(vmcross[vmcross$dist > 0.2 & vmcross$dist<0.4,]$gamma)
mean(vmcross[vmcross$dist > 0.4 & vmcross$dist<0.6,]$gamma)
mean(vmcross[vmcross$dist > 0.6 & vmcross$dist<0.8,]$gamma)
mean(vmcross[vmcross$dist > 0.8 & vmcross$dist<1,]$gamma)
mean(vmcross[vmcross$dist > 1 & vmcross$dist<1.2,]$gamma)
mean(vmcross[vmcross$dist > 1.2 & vmcross$dist<1.4,]$gamma)
mean(vmcross[vmcross$dist > 1.4 & vmcross$dist<1.6,]$gamma)
mean(vmcross[vmcross$dist > 1.6 & vmcross$dist<1.8,]$gamma)
mean(vmcross[vmcross$dist > 1.8 & vmcross$dist<2,]$gamma)
mean(vmcross[vmcross$dist > 2 & vmcross$dist<2.2,]$gamma)
mean(vmcross[vmcross$dist > 2.2,]$gamma)
```

The 0.1-trimmed variogram estimations are computed with

```
y1<-mean(vmcross[vmcross$dist < 0.2,]$gamma,0.1)
y2<-mean(vmcross[vmcross$dist > 0.2 & vmcross$dist<0.4,]$gamma,0.1)
y3<-mean(vmcross[vmcross$dist > 0.4 & vmcross$dist<0.6,]$gamma,0.1)
y4<-mean(vmcross[vmcross$dist > 0.6 & vmcross$dist<0.8,]$gamma,0.1)
y5<-mean(vmcross[vmcross$dist > 0.8 & vmcross$dist<1,]$gamma,0.1)
y6<-mean(vmcross[vmcross$dist > 1 & vmcross$dist<1.2,]$gamma,0.1)
y7<-mean(vmcross[vmcross$dist > 1.2 & vmcross$dist<1.4,]$gamma,0.1)
y8<-mean(vmcross[vmcross$dist > 1.4 & vmcross$dist<1.6,]$gamma,0.1)
y9<-mean(vmcross[vmcross$dist > 1.6 & vmcross$dist<1.8,]$gamma,0.1)
y10<-mean(vmcross[vmcross$dist > 1.8 & vmcross$dist<2,]$gamma,0.1)
y11<-mean(vmcross[vmcross$dist > 2 & vmcross$dist<2.2,]$gamma,0.1)
y12<-mean(vmcross[vmcross$dist > 2.2,]$gamma,0.1)

y<-c(y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12)
```

The Huber's cross-variogram estimations are computed with

```
library(MASS)
u1<-huber(vmcross[vmcross$dist < 0.2,]$gamma)$mu
u2<-huber(vmcross[vmcross$dist > 0.2 & vmcross$dist<0.4,]$gamma)$mu
u3<-huber(vmcross[vmcross$dist > 0.4 & vmcross$dist<0.6,]$gamma)$mu
u4<-huber(vmcross[vmcross$dist > 0.6 & vmcross$dist<0.8,]$gamma)$mu
u5<-huber(vmcross[vmcross$dist > 0.8 & vmcross$dist<1  ,]$gamma)$mu
u6<-huber(vmcross[vmcross$dist > 1   & vmcross$dist<1.2,]$gamma)$mu
u7<-huber(vmcross[vmcross$dist > 1.2 & vmcross$dist<1.4,]$gamma)$mu
u8<-huber(vmcross[vmcross$dist > 1.4 & vmcross$dist<1.6,]$gamma)$mu
u9<-huber(vmcross[vmcross$dist > 1.6 & vmcross$dist<1.8,]$gamma)$mu
u10<-huber(vmcross[vmcross$dist > 1.8 & vmcross$dist<2,]$gamma)$mu
u11<-huber(vmcross[vmcross$dist > 2   &  vmcross$dist<2.2,]$gamma)$mu
u12<-huber(vmcross[vmcross$dist > 2.2,]$gamma)$mu

u<-c(u1,u2,u3,u4,u5,u6,u7,u8,u9,u10,u11,u12)
```

Figure 6 of the paper is obtained with

```
plot(x,vvmcross$gamma,ylim=c(0,2),pch=16,ylab=" ",xlab="h")
points(x,y,pch=16,col=3)
points(x,u,pch=16,col=2)
text(1.7,1.9,"                          *method-of-moments")
text(1.7,1.8,"          *0.1-trimmed",col=3)
text(1.7,1.7,"*Huber",col=2)
```

Classical linearized version:

The classical cross-variogram model is increasing, before the range =1, in the first 5 lags and

si= partial sill + nugget = 1.35202753 + 0.07601742 = 1.428045

Intersection point: (partial sill/slope , si) = (1.35202753/1.680914 , 1.428045) = (0.7523586 , 1.428045)

Thus, the linearized versions are computed with:

```
> x<-seq(0,12*0.2,len=12)
> rectacla<-lm(vvmcross$gamma[1:5]-0.07601742~x[1:5]-1)
> rectacla$coefficients
  x[1:5]
1.680914


  0.07601742+1.680914  * h    ,  h =< 1.35202753/1.680914
  1.428045     ,  h > 1.35202753/1.680914


> clasivarilinemodel<-function(h)
{
 0.07601742+ 1.35202753 *ifelse(h>1.35202753/1.680914,1,1.680914*h/1.35202753)
}
```

Robust 0.1-trimmed linearized version:

```
> rectarecor<-lm(y[1:5]-0.07601742~x[1:5]-1)

> rectarecor$coefficients
  x[1:5]
1.011655

> mean(y[6:12])
[1] 0.9819956


0.07601742+1.011655   * h    ,  h =< (0.9819956-0.07601742)/1.011655  =  0.8955407
0.9819956                      ,  h >  0.8955407


recorvarilinemodel<-function(h)
{
 0.9819956 *ifelse(h>0.8955407,1,(0.07601742+1.011655 *h)/0.9819956 )
}
```

Robust Huber's linearized version:

```
> rectahuber<-lm(u[1:5]-0.07601742~x[1:5]-1)

> rectahuber$coefficients
 x[1:5]
 0.8868434

> mean(u[6:12])
[1] 0.8819796


0.07601742+0.8868434 * h    ,  h =< (0.8819796-0.07601742)/0.8868434  = 0.9087988
0.8819796     ,  h > 0.9087988


hubervarilinemodel<-function(h)
{
  0.8819796  *ifelse(h> 0.9087988,1,(0.07601742+0.8868434 *h)/0.8819796 )
}
```

Classical Spherical Model

```
0.07601742+1.35202753/2 * (3*h-h^3) , h =<1
  1.428045       , h> 1

model22<-function(h){
1.428045*ifelse(h>1,1,(0.07601742+1.35202753/2*(3*h-h^3))/1.428045)
}
```

Figure 9 of the paper is obtained with

```
plot(x,vvmcross$gamma,ylim=c(0,2),pch=16,ylab=" ",xlab="h")
points(x,y,pch=16,col=3)
points(x,u,pch=16,col=2)
text(1.7,1.9,"                        *method-of-moments")
text(1.7,1.8,"          *0.1-trimmed",col=3)
text(1.7,1.7,"*Huber",col=2)
h<-seq(0,2.5,len=10000)
lines(h,clasivarilinemodel(h),type="l",lwd=2)
lines(h,recorvarilinemodel(h),type="l",col=3,lwd=2)
lines(h,hubervarilinemodel(h),type="l",col=2,lwd=2)
lines(h,model22(h),type="l",col=4,lwd=2)
```

From a visual point of view these three linearized versions can be accepted. Using the test considered in Section 7, to obtain **Table 3**:

```
Classical,

x<-seq(0,12*0.2,len=12)

Test Statistic S_n:

> vvmcross
     np       dist     gamma dir.hor dir.ver       id
1   908 0.08644121 0.2573405       0       0 logPb.Ni
2  1844 0.31441297 0.6564834       0       0 logPb.Ni
3  2440 0.49499138 0.9677398       0       0 logPb.Ni
4  3198 0.71534068 0.9456095       0       0 logPb.Ni
5  2914 0.90005368 1.5834326       0       0 logPb.Ni
6  4462 1.09236560 1.6282840       0       0 logPb.Ni
7  4528 1.30215002 2.0084659       0       0 logPb.Ni
8  4932 1.50010567 1.3030284       0       0 logPb.Ni
9  4512 1.70695699 1.2445932       0       0 logPb.Ni
10 4236 1.89091691 1.1766248       0       0 logPb.Ni
11 4512 2.09530679 1.5972814       0       0 logPb.Ni
12  352 2.21278828 1.8455482       0       0 logPb.Ni


> 2*clasivarilinemodel(x)
 [1] 0.1520348 0.8855246 1.6190143 2.3525041 2.8560899 2.8560899 2.8560899
 [8] 2.8560899 2.8560899 2.8560899 2.8560899 2.8560899

> max(abs(2*vvmcross$gamma-2*clasivarilinemodel(x)))
[1] 1.160842
```

1-p-value $= F_{S_n}(v)$. Because N_h is large we use napprox2

```
> vvmcross$np
 [1]  908 1844 2440 3198 2914 4462 4528 4932 4512 4236 4512  352

> 2*dd$gamma
 [1] 0.1802794 0.3086148 0.3213601 0.3339166 0.3299529 0.3469299 0.3639288
 [8] 0.3640443 0.3609931 0.3755922 0.3860178 0.3110649

> 2*nn$gamma
 [1]  30.48875  76.03721  95.06463 119.80589 152.98529 157.71126 178.88583
 [8] 159.21670 179.28215 136.72716 155.15266 167.59549

> 2*vvmcross$gamma
 [1] 0.514681 1.312967 1.935480 1.891219 3.166865 3.256568 4.016932 2.606057
 [9] 2.489186 2.353250 3.194563 3.691096

> max(abs(2*vvmcross$gamma-2*clasivarilinemodel(x)))
[1] 1.160842

> 2*clasivarilinemodel(x)
 [1] 0.1520348 0.8855246 1.6190143 2.3525041 2.8560899 2.8560899 2.8560899
 [8] 2.8560899 2.8560899 2.8560899 2.8560899 2.8560899

> pvalorclasico<-
napprox2(vvmcross$np,2*dd$gamma,2*nn$gamma,2*vvmcross$gamma,-1.160842+
     2*clasivarilinemodel(x),0.01,1.1)-
napprox2(vvmcross$np,2*dd$gamma,2*nn$gamma,2*vvmcross$gamma,1.160842+
     2*clasivarilinemodel(x),0.01,1.1)

> pvalorclasico
 [1] 0.9999923 0.9067626 0.9034902 0.8279266 0.7578355 0.7920466 0.4883016
 [8] 0.8210085 0.7383666 0.7808104 0.7728589 0.3089063

> prod(pvalorclasico)
[1] 0.02246528

Hence, p-value = 1-0.02246528 =  0.9775347.


0.1-trimmed,

Statistics S_n:

> max(abs(2*y-2*recorvarilinemodel(x)))
[1] 0.8783457


> alpha<-0.1
```

```
> ite<-100
> c2<-(1/((1-2*alpha)^(1/(ite+1))))-1
> c1<-((1-2*alpha)^(1/(ite+1)))-1

> corregido
 [1] 1.0000000 0.9999971 0.9999658 0.9792421 0.9840390 0.9958821 0.9998032
 [8] 0.9809274 0.8394706 0.8552197 0.9903931 0.9577881


> pvalorrecortado<-
(corregido-
napprox2(vvmcross$np,2*w,2*m,2*y,0.8783457+2*recorvarilinemodel(x),0.01,1.1) )


> prod(pvalorrecortado)
[1] 0.2742243

p-value:

> 1-prod(pvalorrecortado)
[1] 0.7257757

--------

Huber's cross-variogram



> vvmcross$np
 [1]  908 1844 2440 3198 2914 4462 4528 4932 4512 4236 4512  352

> 2*xt
 [1] 0.07191903 0.18636726 0.18784741 0.19120741 0.18723365 0.21987242
 [7] 0.23636351 0.22916173 0.20185108 0.20187091 0.23349035 0.14125729

> 2*jt
 [1]  14.51452  38.76765  60.33071  85.97760 109.76439 110.68221 138.26014
 [8] 119.19092 127.23367  92.30994  97.38743  80.62086

> 2*u
 [1] 0.1872164 0.5903179 1.0600680 1.0868714 1.7897167 1.8926154 2.6581518
 [8] 1.8137467 1.3144777 1.2573091 1.7631292 1.6482843

saddlepoints in:


> 2*hubervarilinemodel(x)
 [1] 0.1520348 0.5390211 0.9260073 1.3129935 1.6999797 1.7639592 1.7639592
 [8] 1.7639592 1.7639592 1.7639592 1.7639592 1.7639592
```

```
saddle<-function(z0,va,t){
z0<-z0
va<-va
dentroz<-function(x){exp(z0*huber(x-t,1)$mu)*(huber(x-t,1)$mu)*dgamma(x,0.5,2*va)}
solu<-integrate(dentroz,-Inf,Inf)$value
return(solu)
}


> saddle(-0.26,0.1520348,0.8941926)
[1] -0.01174261

> saddle(0.27,0.5390211,0.8941926)
[1] -0.01567544

> saddle(0.55,0.9260073,0.8941926)
[1] 0.07556215

> saddle(0.6,1.3129935,0.8941926)
[1] -0.09518467

> saddle(0.9,1.6999797,0.8941926)
[1] -0.02683211

> saddle(1,1.7639592,0.8941926)
[1] 0.04978794
```

Statistics S_n:

```
> max(abs(2*u-2*hubervarilinemodel(x)))
[1] 0.8941926
```

```
library(MASS)

napproxhube2r<-function(n,rhoi,rhoj,gammaij,t,e,g,z0,b)
{
rho<-gammaij/(rhoi*rhoj)
dentrog<-function(x){exp(z0*huber(x-t,1)$mu)*(1/(pi*sqrt(1-rho^2)))*
       exp((x/(g^2*rhoi*rhoj))*rho/(1-rho^2))*
```

```
besselK(abs(x/(g^2*rhoi*rhoj))/(1-rho^2),0)}
Bg1<-integrate(dentrog,-3,0)$value
Bg2<-integrate(dentrog,0,3)$value
Bg<-Bg1+Bg2
dentro1<-function(x){exp(z0*huber(x-t,1)$mu)*(1/(pi*sqrt(1-rho^2)))*
       exp((x/(rhoi*rhoj))*rho/(1-rho^2))*
besselK(abs(x/(g^2*rhoi*rhoj))/(1-rho^2),0)}
B11<-integrate(dentro1,-3,0)$value
B12<-integrate(dentro1,0,3)$value
B1<-B11+B12
s<-sqrt(abs(-2*n*log(B1)))
dentro2<-function(x){exp(z0*huber(x-t,1)$mu)*((huber(x-t,1)$mu)^2)*(1/(pi*sqrt(1-rho^2)))*
       exp((x/(rhoi*rhoj))*rho/(1-rho^2))*
besselK(abs(x/(g^2*rhoi*rhoj))/(1-rho^2),0)}
B21<-integrate(dentro2,-3,0)$value
B22<-integrate(dentro2,0,3)$value
B2<-B21+B22
r1<-z0*sqrt(B2)
r<-sqrt(n)*r1
A<-1-pnorm(s)+dnorm(s)*(1/r+1/s)
resul<-A+e*sqrt(n)*dnorm(s)/r1*((Bg/B1)-1)
return(resul)
}
```

1-p-value $= F_{S_n}(v)$:

```
library(MASS)

(  napproxhube2r(908,0.07191903,14.51452,0.1872164,  -0.8941926+0.1520348, 0.01,1.1, -0.01174261,1)-
napproxhube2r(908,0.07191903,14.51452,0.1872164,  0.8941926+2*0.1520348, 0.01,1.1, -0.01174261,1)  )*
(  napproxhube2r(1844,0.18636726,38.76765, 0.5903179,  -0.8941926+ 0.5390211, 0.01,1.1, -0.01567544,1)-
napproxhube2r(1844,0.18636726,38.76765, 0.5903179,  0.8941926+ 0.5390211, 0.01,1.1, -0.01567544,1)  )*
(  napproxhube2r(2440,0.18784741, 60.33071 , 1.0600680,  -0.8941926+ 0.9260073, 0.01,1.1, 0.07556215,1)-
napproxhube2r(2440,0.18784741, 60.33071 , 1.0600680,  0.8941926+ 0.9260073, 0.01,1.1, 0.07556215,1)  )*
(napproxhube2r(3198, 0.19120741,  85.97760 , 1.0868714,  -0.8941926+ 1.3129935, 0.01,1.1, -0.09518467,1)-
napproxhube2r(3198, 0.19120741,  85.97760 , 1.0868714,  0.8941926+ 1.3129935, 0.01,1.1, -0.09518467,1) )*
(napproxhube2r(2914, 0.18723365,  109.76439 , 1.7897167,  -0.8941926+ 1.6999797, 0.01,1.1, -0.02683211,1)-
napproxhube2r(2914, 0.18723365,  109.76439 , 1.7897167,  0.8941926+ 1.6999797, 0.01,1.1, -0.02683211,1) )*
(napproxhube2r(4462,  0.21987242,  110.68221 , 1.8926154,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(4462,  0.21987242,  110.68221 , 1.8926154,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1))*
(napproxhube2r(4528,  0.23636351 , 138.26014 , 2.6581518,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(4528,  0.23636351 , 138.26014 , 2.6581518,  0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1))*
( napproxhube2r(4932,0.22916173,119.19092 , 1.8137467,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(4932,0.22916173,119.19092 , 1.8137467,  0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1) )*
(napproxhube2r(4512,0.20185108,127.23367 , 1.3144777,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(4512,0.20185108,127.23367 , 1.3144777,  0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1))*
(napproxhube2r(4236,0.20187091,92.30994 ,1.2573091,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(4236,0.20187091,92.30994 ,1.2573091,  0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1))*
(napproxhube2r(4512,0.23349035,  97.38743 ,1.7631292,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(4512,0.23349035,  97.38743 ,1.7631292,  0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1) )*
(  napproxhube2r(352,0.14125729, 80.62086 ,1.6482843,  -0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)-
napproxhube2r(352,0.14125729, 80.62086 ,1.6482843,  0.8941926+ 1.7639592, 0.01,1.1, 0.04978794,1)  )
[1]  0
```

Hence, p-value = 1.

Variograms of the two variables (García-Pérez, 2020)

Let us see if we can accept the linear variograms for the variables:

log(Pb)

```
library(gstat)
ddd<-variogram(log(prediction.dat$Pb)~1, prediction.dat,cloud=T,width=0.2)

dd<-variogram(log(prediction.dat$Pb)~1, prediction.dat,width=0.2)

> dd
     np       dist      gamma dir.hor dir.ver   id
1   454 0.08644121 0.09013972       0       0 var1
2   922 0.31441297 0.15430741       0       0 var1
3  1220 0.49499138 0.16068005       0       0 var1
4  1599 0.71534068 0.16695832       0       0 var1
5  1457 0.90005368 0.16497647       0       0 var1
6  2231 1.09236560 0.17346496       0       0 var1
7  2264 1.30215002 0.18196439       0       0 var1
8  2466 1.50010567 0.18202214       0       0 var1
9  2256 1.70695699 0.18049656       0       0 var1
10 2118 1.89091691 0.18779612       0       0 var1
11 2256 2.09530679 0.19300891       0       0 var1
12  176 2.21278828 0.15553244       0       0 var1

------------

Matheron's variogram estimations:

mean(ddd[ddd$dist < 0.2,]$gamma)
mean(ddd[ddd$dist > 0.2 & ddd$dist<0.4,]$gamma)
mean(ddd[ddd$dist > 0.4 & ddd$dist<0.6,]$gamma)
mean(ddd[ddd$dist > 0.6 & ddd$dist<0.8,]$gamma)
mean(ddd[ddd$dist > 0.8 & ddd$dist<1.0,]$gamma)
mean(ddd[ddd$dist > 1.0 & ddd$dist<1.2,]$gamma)
mean(ddd[ddd$dist > 1.2 & ddd$dist<1.4,]$gamma)
mean(ddd[ddd$dist > 1.4 & ddd$dist<1.6,]$gamma)
mean(ddd[ddd$dist > 1.6 & ddd$dist<1.8,]$gamma)
mean(ddd[ddd$dist > 1.8 & ddd$dist<2.0,]$gamma)
mean(ddd[ddd$dist > 2.0 & ddd$dist<2.2,]$gamma)
mean(ddd[ddd$dist > 2.2,]$gamma)

> mean(ddd[ddd$dist < 0.2,]$gamma)
[1] 0.09013972
> mean(ddd[ddd$dist > 0.2 & ddd$dist<0.4,]$gamma)
[1] 0.1543074
> mean(ddd[ddd$dist > 0.4 & ddd$dist<0.6,]$gamma)
[1] 0.16068
```

```
> mean(ddd[ddd$dist > 0.6 & ddd$dist<0.8,]$gamma)
[1] 0.1669583
> mean(ddd[ddd$dist > 0.8 & ddd$dist<1.0,]$gamma)
[1] 0.1649765
> mean(ddd[ddd$dist > 1.0 & ddd$dist<1.2,]$gamma)
[1] 0.173465
> mean(ddd[ddd$dist > 1.2 & ddd$dist<1.4,]$gamma)
[1] 0.1819644
> mean(ddd[ddd$dist > 1.4 & ddd$dist<1.6,]$gamma)
[1] 0.1820221
> mean(ddd[ddd$dist > 1.6 & ddd$dist<1.8,]$gamma)
[1] 0.1804966
> mean(ddd[ddd$dist > 1.8 & ddd$dist<2.0,]$gamma)
[1] 0.1877961
> mean(ddd[ddd$dist > 2.0 & ddd$dist<2.2,]$gamma)
[1] 0.1930089
> mean(ddd[ddd$dist > 2.2,]$gamma)
[1] 0.1555324

> dd$gamma
 [1] 0.09013972 0.15430741 0.16068005 0.16695832 0.16497647 0.17346496
 [7] 0.18196439 0.18202214 0.18049656 0.18779612 0.19300891 0.15553244

----------

0.1-trimmed variogram estimations:


w1<-mean(ddd[ddd$dist < 0.2,]$gamma,0.1)
w2<-mean(ddd[ddd$dist > 0.2 & ddd$dist<0.4,]$gamma,0.1)
w3<-mean(ddd[ddd$dist > 0.4 & ddd$dist<0.6,]$gamma,0.1)
w4<-mean(ddd[ddd$dist > 0.6 & ddd$dist<0.8,]$gamma,0.1)
w5<-mean(ddd[ddd$dist > 0.8 & ddd$dist<1.0,]$gamma,0.1)
w6<-mean(ddd[ddd$dist > 1.0 & ddd$dist<1.2,]$gamma,0.1)
w7<-mean(ddd[ddd$dist > 1.2 & ddd$dist<1.4,]$gamma,0.1)
w8<-mean(ddd[ddd$dist > 1.4 & ddd$dist<1.6,]$gamma,0.1)
w9<-mean(ddd[ddd$dist > 1.6 & ddd$dist<1.8,]$gamma,0.1)
w10<-mean(ddd[ddd$dist > 1.8 & ddd$dist<2.0,]$gamma,0.1)
w11<-mean(ddd[ddd$dist > 2.0 & ddd$dist<2.2,]$gamma,0.1)
w12<-mean(ddd[ddd$dist > 2.2,]$gamma,0.1)


w<-c(w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w2)

> w
 [1] 0.04458231 0.09911849 0.10172569 0.10368365 0.10878252 0.11716422
 [7] 0.12624456 0.12254889 0.11311406 0.11277025 0.12905951 0.09911849
```

Huber's variogram estimations:

```
library(MASS)
x1<-huber(ddd[ddd$dist < 0.2,]$gamma)$mu
x2<-huber(ddd[ddd$dist > 0.2 & ddd$dist<0.4,]$gamma)$mu
x3<-huber(ddd[ddd$dist > 0.4 & ddd$dist<0.6,]$gamma)$mu
x4<-huber(ddd[ddd$dist > 0.6 & ddd$dist<0.8,]$gamma)$mu
x5<-huber(ddd[ddd$dist > 0.8 & ddd$dist<1.0,]$gamma)$mu
x6<-huber(ddd[ddd$dist > 1.0 & ddd$dist<1.2,]$gamma)$mu
x7<-huber(ddd[ddd$dist > 1.2 & ddd$dist<1.4,]$gamma)$mu
x8<-huber(ddd[ddd$dist > 1.4 & ddd$dist<1.6,]$gamma)$mu
x9<-huber(ddd[ddd$dist > 1.6 & ddd$dist<1.8,]$gamma)$mu
x10<-huber(ddd[ddd$dist > 1.8 & ddd$dist<2.0,]$gamma)$mu
x11<-huber(ddd[ddd$dist > 2.0 & ddd$dist<2.2,]$gamma)$mu
x12<-huber(ddd[ddd$dist > 2.2,]$gamma)$mu

xt<-c(x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12)

> xt
 [1] 0.03595952 0.09318363 0.09392371 0.09560371 0.09361683 0.10993621
 [7] 0.11818175 0.11458087 0.10092554 0.10093546 0.11674517 0.07062865

---

x<-seq(0,12*0.2,len=12)
plot(x,dd$gamma,ylim=c(0,0.2),pch=16,ylab="semivariogram",xlab="h")
points(x,w,pch=16,col=3)
points(x,xt,pch=16,col=2)
text(1.5,0.05,"     *Matheron")
text(1.5,0.03,"           *0.1-trimmed",col=3)
text(1.5,0.01,"*Huber",col=2)


---


> plot(variogram(log(prediction.dat$Pb)~1,prediction.dat,cloud=F,width=0.2),pch=16)
```

Initially we consider the parameters:

nugget=0, range=1, partial sill=0.18

```
> esti1<- variogram(log(prediction.dat$Pb)~1,prediction.dat,width=0.2)
> model1<-vgm(0.18,"Sph",1,0)
> plot(esti1,model1,pch=16)

> fit.variogram(esti1,model1)
  model      psill     range
```

```
1   Nug 0.05992532 0.0000000
2   Sph 0.11011524 0.4671544
```

Now, the parameters we are going to consider are:

nugget=0.05992532,range=0.4671544,partial sill=0.11011524,sill=0.11011524+0.05992532=0.1700406

```
model11<-vgm(0.11011524,"Sph",0.4671544,0.05992532)
plot(esti1,model11,pch=16)
```

We are going to define the linearized variogram model. Because there are three values before range=0.4671544

Classical linearized version:

```
rectacla1<-lm(dd$gamma[1:3]-0.05992532~x[1:3]-1)
> rectacla1$coefficients
   x[1:3]
0.2712339

0.05992532 + 0.2712339 * h   ,  h =< 0.11011524/0.2712339 =  0.4059789
0.1700406    ,  h >  0.4059789

clasivarilinemodel1<-function(h)
{
  0.1700406*ifelse(h>0.4059789,1,(0.05992532 + 0.2712339 * h) /0.1700406)
}
```

0.1-trimmed linearized version:

```
rectarecor1<-lm(w[1:3]-0.05992532~x[1:3]-1)

> rectarecor1$coefficients
   x[1:3]
0.1125611

> mean(w[4:12])
[1] 0.1147207


0.05992532+0.1125611    * h   ,  h =< (0.1147207-0.05992532)/0.1125611 = 0.4868057
  0.1147207    ,  h > 0.4868057
```

```
recorvarilinemodel1<-function(h)
{
 0.1147207*ifelse(h>0.4868057,1,(0.05992532+0.1125611* h )/ 0.1147207 )
}

----

Huber's linearized version:

rectahuber1<-lm(xt[1:3]-0.05992532~x[1:3]-1)

> rectahuber1$coefficients
  x[1:3]
0.09281716

> mean(xt[4:12])
[1] 0.1023505


0.05992532 + 0.09281716  * h    ,  h =< (0.1023505-0.05992532)/0.09281716  = 0.4570834
0.1023505    ,  h > 0.4570834

hubervarilinemodel1<-function(h)
{
  0.1023505 *ifelse(h>0.4570834,1,(0.05992532 + 0.09281716  * h)/0.1023505 )
}
```

Figure 7 of the paper is obtained with

```
x<-seq(0,12*0.2,len=12)
plot(x,dd$gamma,ylim=c(0,0.22),pch=16,ylab="semivariogram",xlab="h",main="log(Lead)")
points(x,w,pch=16,col=3)
points(x,xt,pch=16,col=2)

h<-seq(0,12*0.2,len=10000)
lines(h,clasivarilinemodel1(h),type="l")
lines(h,recorvarilinemodel1(h),type="l",col=3)
lines(h,hubervarilinemodel1(h),type="l",col=2)
text(1.5,0.05,"     *Matheron")
text(1.5,0.03,"           *0.1-trimmed",col=3)
text(1.5,0.01,"*Huber",col=2)


----------


Global tests:


Classical,

x<-seq(0,12*0.2,len=12)
dd

> dd
     np       dist      gamma dir.hor dir.ver   id
1   454 0.08644121 0.09013972       0       0 var1
2   922 0.31441297 0.15430741       0       0 var1
3  1220 0.49499138 0.16068005       0       0 var1
4  1599 0.71534068 0.16695832       0       0 var1
5  1457 0.90005368 0.16497647       0       0 var1
6  2231 1.09236560 0.17346496       0       0 var1
7  2264 1.30215002 0.18196439       0       0 var1
8  2466 1.50010567 0.18202214       0       0 var1
9  2256 1.70695699 0.18049656       0       0 var1
10 2118 1.89091691 0.18779612       0       0 var1
11 2256 2.09530679 0.19300891       0       0 var1
12  176 2.21278828 0.15553244       0       0 var1
```

```
Classical statistics:

     Nh      dist  2*hat{gamma(h)}  2*gamma(h)  2*hat{gamma(h)}-2*gamma(h)  sup |2*hat{gamma(h)}-2*gamma(h)|
1   454  0.08644121   0.1802794     0.1198506        0.0604288
2   922  0.31441297   0.3086148     0.2382073        0.0704075  -------------------------->   0.0704075
3  1220  0.49499138   0.3213601     0.3400812       -0.0187211
4  1599  0.71534068   0.3339166     0.3400812       -0.0061646
5  1457  0.90005368   0.3299529     0.3400812       -0.0101283
6  2231  1.09236560   0.3469299     0.3400812        0.0068487
7  2264  1.30215002   0.3639288     0.3400812        0.0238476
8  2466  1.50010567   0.3640443     0.3400812        0.0239631
9  2256  1.70695699   0.3609931     0.3400812        0.0209119
10 2118  1.89091691   0.3755922     0.3400812        0.035511
11 2256  2.09530679   0.3860178     0.3400812        0.0459366
12  176  2.21278828   0.3110649     0.3400812       -0.0290163


2*hat{gamma(h)}  is (twice) the value of the classical Matheron's estimator in the x abscise
(of    x<-seq(0,12*0.2,len=12))    given by   2*dd$gamma:

> 2*dd$gamma
 [1] 0.1802794 0.3086148 0.3213601 0.3339166 0.3299529 0.3469299 0.3639288
 [8] 0.3640443 0.3609931 0.3755922 0.3860178 0.3110649

2*gamma(h) is (twice) the value of the Classical linearized version in the x abscise
(of    x<-seq(0,12*0.2,len=12))    given by   2*clasivarilinemodel1.

> 2*clasivarilinemodel1(x)
 [1] 0.1198506 0.2382073 0.3400812 0.3400812 0.3400812 0.3400812 0.3400812
 [8] 0.3400812 0.3400812 0.3400812 0.3400812 0.3400812

Statistics S_n:

> max(abs(2*dd$gamma-2*clasivarilinemodel1(x)))
[1] 0.07040758
```

1-p-value $= F_{S_n}(v)$:

```
(approx(0.01,454,-0.0704076+0.1198506,0.1198506,1.1)-approx(0.01,454,0.0704076+0.1198506,0.1198506,1.1))*
(approx(0.01,922,-0.0704076+0.2382073,0.2382073,1.1)-approx(0.01,922,0.0704076+0.2382073,0.2382073,1.1))*
(approx(0.01,1220,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,1220,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,1599,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,1599,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,1457,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,1457,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,2231,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,2231,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,2264,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,2264,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,2466,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,2466,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,2256,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,2256,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,2118,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,2118,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,2256,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,2256,0.0704076+0.3400812,0.3400812,1.1))*
(approx(0.01,176,-0.0704076+0.3400812,0.3400812,1.1)-approx(0.01,176,0.0704076+0.3400812,0.3400812,1.1))
[1] 0.9479127
```

Hence, p-value = 1-0.9479127 = 0.0520873

```
0.1-trimmed,

    Nh  dist  2*hat_T{gamma(h)} 2*gamma(h) 2*hat_T{gamma(h)}-2*gamma(h) sup|2*hat_T{gamma(h)}-2*gamma(h)|
1   454  0.08644121   0.08916461     0.1198506    -0.03068599
2   922  0.31441297   0.19823698     0.1689682     0.02926878
3  1220  0.49499138   0.20345138     0.2180858    -0.01463442
4  1599  0.71534068   0.20736730     0.2294414     -0.0220741
5  1457  0.90005368   0.21756505     0.2294414    -0.01187635
6  2231  1.09236560   0.23432844     0.2294414     0.00488704
7  2264  1.30215002   0.25248911     0.2294414      0.0230477
8  2466  1.50010567   0.24509777     0.2294414     0.01565637
9  2256  1.70695699   0.22622811     0.2294414    -0.00321329
10 2118  1.89091691   0.22554051     0.2294414    -0.00390089
11 2256  2.09530679   0.25811903     0.2294414     0.02867763
12  176  2.21278828   0.19823698     0.2294414    -0.03120442      -------------------->  0.03120442


2*hat_T{gamma(h)} is (twice) the value of the trimmed estimator in the x abscise
(of    x<-seq(0,12*0.2,len=12))    given by   2*w:

2*hat_T{gamma(h)} = 2*w

> 2*w
 [1] 0.08916461 0.19823698 0.20345138 0.20736730 0.21756505 0.23432844
 [7] 0.25248911 0.24509777 0.22622811 0.22554051 0.25811903 0.19823698


2*gamma(h) is (twice) the value of the  0.1-trimmed linearized version in the x abscise
(of    x<-seq(0,12*0.2,len=12))    given by   2*recorvarilinemodel1

> 2*recorvarilinemodel1(x)
 [1] 0.1198506 0.1689682 0.2180858 0.2294414 0.2294414 0.2294414 0.2294414
 [8] 0.2294414 0.2294414 0.2294414 0.2294414 0.2294414


Statistics S_n:

> max(abs(2*w-2*recorvarilinemodel1(x)))
[1] 0.03120442
```

1-p-value $= F_{S_n}(v)$:

```
alpha<-0.1
ite<-20000
c2<-(1/((1-2*alpha)^(1/(ite+1))))-1
c1<-((1-2*alpha)^(1/(ite+1)))-1
(((1+454*c1)^(ite+1))*((1+454*c2)^(ite+1))*(approx(0.01,454,-0.03120442+0.1198506,0.1198506,1.1)
-approx(0.01,454,0.03120442+0.1198506,0.1198506,1.1)))*
(((1+922*c1)^(ite+1))*((1+922*c2)^(ite+1))*(approx(0.01,922,-0.03120442+0.1689682,0.1689682,1.1)
-approx(0.01,922,0.03120442+0.1689682,0.1689682,1.1)))*
(((1+1220*c1)^(ite+1))*((1+1220*c2)^(ite+1))*(approx(0.01,1220,-0.03120442+0.2180858,0.2180858,1.1)
-approx(0.01,1220,0.03120442+0.2180858,0.2180858 ,1.1)))*
(((1+1599*c1)^(ite+1))*((1+1599*c2)^(ite+1))*(approx(0.01,1599,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,1599,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+1457*c1)^(ite+1))*((1+1457*c2)^(ite+1))*(approx(0.01,1457,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,1457,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+2231*c1)^(ite+1))*((1+2231*c2)^(ite+1))*(approx(0.01,2231,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,2231,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+2264*c1)^(ite+1))*((1+2264*c2)^(ite+1))*(approx(0.01,2264,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,2264,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+2466*c1)^(ite+1))*((1+2466*c2)^(ite+1))*(approx(0.01,2466,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,2466,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+2256*c1)^(ite+1))*((1+2256*c2)^(ite+1))*(approx(0.01,2256,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,2256,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+2118*c1)^(ite+1))*((1+2118*c2)^(ite+1))*(approx(0.01,2118,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,2118,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+2256*c1)^(ite+1))*((1+2256*c2)^(ite+1))*(approx(0.01,2256,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,2256,0.03120442+0.2294414,0.2294414,1.1)))*
(((1+176*c1)^(ite+1))*((1+176*c2)^(ite+1))*(approx(0.01,176,-0.03120442+0.2294414,0.2294414,1.1)
-approx(0.01,176,0.03120442+0.2294414,0.2294414,1.1)))
[1] 5.164723e-42
```

```
p-value:

> 1- 5.164723e-42
[1] 1
```

Huber,

```
     Nh  dist  2*hat_H{gamma(h)}  2*gamma(h) 2*hat_T{gamma(h)}-2*gamma(h)  sup|2*hat_T{gamma(h)}-2*gamma(h)|
1   454  0.08644121   0.07191903      0.1198506    -0.04793157
2   922  0.31441297   0.18636726      0.1603527     0.02601456
3  1220  0.49499138   0.18784741      0.2008547    -0.01300729
4  1599  0.71534068   0.19120741      0.2047010    -0.01349359
5  1457  0.90005368   0.18723365      0.2047010    -0.01746735
6  2231  1.09236560   0.21987242      0.2047010     0.01517142
7  2264  1.30215002   0.23636351      0.2047010     0.03166251
8  2466  1.50010567   0.22916173      0.2047010     0.02446073
9  2256  1.70695699   0.20185108      0.2047010    -0.00284992
10 2118  1.89091691   0.20187091      0.2047010    -0.00283009
11 2256  2.09530679   0.23349035      0.2047010     0.02878935
12  176  2.21278828   0.14125729      0.2047010    -0.06344371 ----------------->  0.06344371
```

```
2*hat_H{gamma(h)} is (twice) the value of the Huber's estimator in the x abscise
(of    x<-seq(0,12*0.2,len=12))    given by   2*xt:

2*hat_H{gamma(h)} = 2*xt

> 2*xt
 [1] 0.07191903 0.18636726 0.18784741 0.19120741 0.18723365 0.21987242
 [7] 0.23636351 0.22916173 0.20185108 0.20187091 0.23349035 0.14125729

2*gamma(h) is (twice) the value of the Huber's linearized version in the x abscise
(of    x<-seq(0,12*0.2,len=12))    given by   2*hubervarilinemodel1

> 2*hubervarilinemodel1(x)
 [1] 0.1198506 0.1603527 0.2008547 0.2047010 0.2047010 0.2047010 0.2047010
 [8] 0.2047010 0.2047010 0.2047010 0.2047010 0.2047010

Statistics S_n:

> max(abs(2*xt-2*hubervarilinemodel1(x)))
[1] 0.06344371

-----

saddle<-function(z0,va,t){
z0<-z0
va<-va
dentroz<-function(x){exp(z0*huber(x-t,1)$mu)*(huber(x-t,1)$mu)*dgamma(x,0.5,2*va)}
solu<-integrate(dentroz,-Inf,Inf)$value
return(solu)
}

> saddle(-3.102103,0.06531285,0.0711932514)
```

```
[1] 0.01885769

> saddle(-1.64939,0.1306257,0.0711932514)
[1] 0.07987374

> saddle(-1.533117,0.13919836,0.0711932514)
[1] 0.09031659
```

Hence, the z0 of 2gamma(h)=0.06531285 is -3.102103.

The z0 of 2gamma(h)=0.1306257 is -1.64939.

The z0 of 2gamma(h)=0.13919836 is -1.533117.

VOM+SAD approximation (13):

```
approxhuber<-function(e,n,t,va,g,z0,b)
{
dentrog<-function(x){exp(z0*huber(x-t,1)$mu)*dgamma(x,0.5,2*(g^2)*va)}
Bg<-integrate(dentrog,-Inf,Inf)$value
dentro1<-function(x){exp(z0*huber(x-t,1)$mu)*dgamma(x,0.5,2*va)}
B1<-integrate(dentro1,-Inf,Inf)$value
s<-sqrt(-2*n*log(B1))
dentro2<-function(x){exp(z0*huber(x-t,1)$mu)*((huber(x-t,1)$mu)^2)*dgamma(x,0.5,2*va)}
B2<-integrate(dentro2,-Inf,Inf)$value
r1<-z0*sqrt(B2)
r<-sqrt(n)*r1
A<-1-pnorm(s)+dnorm(s)*(1/r+1/s)
resul<-A+e*sqrt(n)*dnorm(s)/r1*((Bg/B1)-1)
return(resul)
}
```

```
> saddle(-4,0.1198506,0.06344371)
[1] 0.01598997

> saddle(-4,0.1603527,0.06344371)
[1] 0.01761052

> saddle(-4,0.2008547,0.06344371)
[1] 0.0187645

> saddle(-4,0.2047010,0.06344371)
[1] 0.01885501
```

Hence, the z0 of 2gamma(h)=0.1198506 is -4. The z0 of 2gamma(h)=0.1603527 is -4.

The z0 of 2gamma(h)=0.2008547 is -4. The z0 of 2gamma(h)=0.2047010 is -4.

1-p-value:

```
library(MASS)
(approxhuber(0.01,454,-0.06344371+0.1198506,0.1198506,1.1,-4,1)
-approxhuber(0.01,454,0.06344371+0.1198506,0.1198506,1.1,-4,1))*
(approxhuber(0.01,922,-0.06344371+0.1603527,0.1603527,1.1,-4,1)
-approxhuber(0.01,922,0.06344371+0.1603527,0.1603527,1.1,-4,1))*
(approxhuber(0.01,1220,-0.06344371+0.2008547,0.2008547,1.1,-4,1)
-approxhuber(0.01,1220,0.06344371+0.2008547,0.2008547,1.1,-4,1))*
(approxhuber(0.01,1599,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,1599,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,1457,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,1457,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,2231,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,2231,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,2264,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,2264,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,2466,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,2466,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,2256,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,2256,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,2118,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,2118,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,2256,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,2256,0.06344371+0.2047010,0.2047010,1.1,-4,1))*
(approxhuber(0.01,176,-0.06344371+0.2047010,0.2047010,1.1,-4,1)
-approxhuber(0.01,176,0.06344371+0.2047010,0.2047010,1.1,-4,1))
[1] 0
```

Hence, p-value = 1.

Ni

```
library(gstat)
library(sp)
data(jura)
coordinates(prediction.dat) = ~Xloc+Yloc

nnn<-variogram(prediction.dat$Ni~1, prediction.dat,cloud=T,width=0.2)

nn<-variogram(prediction.dat$Ni~1, prediction.dat,width=0.2)

> nn
     np       dist    gamma dir.hor dir.ver   id
1   454 0.08644121 15.24437       0       0 var1
2   922 0.31441297 38.01861       0       0 var1
3  1220 0.49499138 47.53232       0       0 var1
4  1599 0.71534068 59.90295       0       0 var1
5  1457 0.90005368 76.49265       0       0 var1
6  2231 1.09236560 78.85563       0       0 var1
7  2264 1.30215002 89.44291       0       0 var1
8  2466 1.50010567 79.60835       0       0 var1
9  2256 1.70695699 89.64108       0       0 var1
10 2118 1.89091691 68.36358       0       0 var1
11 2256 2.09530679 77.57633       0       0 var1
12  176 2.21278828 83.79775       0       0 var1


-----

Matheron's variogram estimations:

mean(nnn[nnn$dist < 0.2,]$gamma)
mean(nnn[nnn$dist > 0.2 & nnn$dist<0.4,]$gamma)
mean(nnn[nnn$dist > 0.4 & nnn$dist<0.6,]$gamma)
mean(nnn[nnn$dist > 0.6 & nnn$dist<0.8,]$gamma)
mean(nnn[nnn$dist > 0.8 & nnn$dist<1.0,]$gamma)
mean(nnn[nnn$dist > 1.0 & nnn$dist<1.2,]$gamma)
mean(nnn[nnn$dist > 1.2 & nnn$dist<1.4,]$gamma)
mean(nnn[nnn$dist > 1.4 & nnn$dist<1.6,]$gamma)
mean(nnn[nnn$dist > 1.6 & nnn$dist<1.8,]$gamma)
mean(nnn[nnn$dist > 1.8 & nnn$dist<2.0,]$gamma)
mean(nnn[nnn$dist > 2.0 & nnn$dist<2.2,]$gamma)
mean(nnn[nnn$dist > 2.2,]$gamma)

> mean(nnn[nnn$dist < 0.2,]$gamma)
[1] 15.24437
> mean(nnn[nnn$dist > 0.2 & nnn$dist<0.4,]$gamma)
[1] 38.01861
```

```
> mean(nnn[nnn$dist > 0.4 & nnn$dist<0.6,]$gamma)
[1] 47.53232
> mean(nnn[nnn$dist > 0.6 & nnn$dist<0.8,]$gamma)
[1] 59.90295
> mean(nnn[nnn$dist > 0.8 & nnn$dist<1.0,]$gamma)
[1] 76.49265
> mean(nnn[nnn$dist > 1.0 & nnn$dist<1.2,]$gamma)
[1] 78.85563
> mean(nnn[nnn$dist > 1.2 & nnn$dist<1.4,]$gamma)
[1] 89.44291
> mean(nnn[nnn$dist > 1.4 & nnn$dist<1.6,]$gamma)
[1] 79.60835
> mean(nnn[nnn$dist > 1.6 & nnn$dist<1.8,]$gamma)
[1] 89.64108
> mean(nnn[nnn$dist > 1.8 & nnn$dist<2.0,]$gamma)
[1] 68.36358
> mean(nnn[nnn$dist > 2.0 & nnn$dist<2.2,]$gamma)
[1] 77.57633
> mean(nnn[nnn$dist > 2.2,]$gamma)
[1] 83.79775

> nn$gamma
 [1] 15.24437 38.01861 47.53232 59.90295 76.49265 78.85563 89.44291 79.60835
 [9] 89.64108 68.36358 77.57633 83.79775

-----

0.1-trimmed variogram estimations:

m1<-mean(nnn[nnn$dist < 0.2,]$gamma,0.1)
m2<-mean(nnn[nnn$dist > 0.2 & nnn$dist<0.4,]$gamma,0.1)
m3<-mean(nnn[nnn$dist > 0.4 & nnn$dist<0.6,]$gamma,0.1)
m4<-mean(nnn[nnn$dist > 0.6 & nnn$dist<0.8,]$gamma,0.1)
m5<-mean(nnn[nnn$dist > 0.8 & nnn$dist<1.0,]$gamma,0.1)
m6<-mean(nnn[nnn$dist > 1.0 & nnn$dist<1.2,]$gamma,0.1)
m7<-mean(nnn[nnn$dist > 1.2 & nnn$dist<1.4,]$gamma,0.1)
m8<-mean(nnn[nnn$dist > 1.4 & nnn$dist<1.6,]$gamma,0.1)
m9<-mean(nnn[nnn$dist > 1.6 & nnn$dist<1.8,]$gamma,0.1)
m10<-mean(nnn[nnn$dist > 1.8 & nnn$dist<2.0,]$gamma,0.1)
m11<-mean(nnn[nnn$dist > 2.0 & nnn$dist<2.2,]$gamma,0.1)
m12<-mean(nnn[nnn$dist > 2.2,]$gamma,0.1)

m<-c(m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m2)

> m
 [1]  8.46811 22.89786 32.22465 42.86607 55.57943 56.12061 66.49540 59.61961
 [9] 63.51613 48.31428 51.71628 22.89786

------
```

```
Huber's variogram estimations:

library(MASS)
j1<-huber(nnn[nnn$dist < 0.2,]$gamma)$mu
j2<-huber(nnn[nnn$dist > 0.2 & nnn$dist<0.4,]$gamma)$mu
j3<-huber(nnn[nnn$dist > 0.4 & nnn$dist<0.6,]$gamma)$mu
j4<-huber(nnn[nnn$dist > 0.6 & nnn$dist<0.8,]$gamma)$mu
j5<-huber(nnn[nnn$dist > 0.8 & nnn$dist<1.0,]$gamma)$mu
j6<-huber(nnn[nnn$dist > 1.0 & nnn$dist<1.2,]$gamma)$mu
j7<-huber(nnn[nnn$dist > 1.2 & nnn$dist<1.4,]$gamma)$mu
j8<-huber(nnn[nnn$dist > 1.4 & nnn$dist<1.6,]$gamma)$mu
j9<-huber(nnn[nnn$dist > 1.6 & nnn$dist<1.8,]$gamma)$mu
j10<-huber(nnn[nnn$dist > 1.8 & nnn$dist<2.0,]$gamma)$mu
j11<-huber(nnn[nnn$dist > 2.0 & nnn$dist<2.2,]$gamma)$mu
j12<-huber(nnn[nnn$dist > 2.2,]$gamma)$mu


jt<-c(j1,j2,j3,j4,j5,j6,j7,j8,j9,j10,j11,j12)

> jt
 [1]  7.257258 19.383823 30.165353 42.988800 54.882194 55.341103 69.130072
 [8] 59.595462 63.616835 46.154969 48.693715 40.310432


----------

x<-seq(0,12*0.2,len=12)
plot(x,nn$gamma,pch=16,ylab="semivariogram",xlab="h",main="Nickel")
points(x,m,pch=16,col=3)
points(x,jt,pch=16,col=2)
text(1.3,30,"     *Matheron")
text(1.3,25,"           *0.1-trimmed",col=3)
text(1.3,20,"*Huber",col=2)


> plot(variogram(prediction.dat$Ni~1,prediction.dat,cloud=F,width=0.2),pch=16)

nugget=0, range=1, partial sill=sill=80

esti2<- variogram(prediction.dat$Ni~1,prediction.dat,width=0.2)
model2<-vgm(80,"Sph",1,0)
plot(esti2,model2,pch=16)

fit.variogram(esti2,model2)

> fit.variogram(esti2,model2)
  model     psill    range
1   Nug  7.943094 0.000000
2   Sph 74.333216 1.276398
```

```
nugget=7.943094, range=1.276398, partial sill=74.333216,
sill=74.333216+7.943094= 82.27631


model21<-vgm(74.333216,"Sph",1.276398,7.943094)
plot(esti2,model21,pch=16)


------

6 values before range=1.276398. We can see it with

> esti2
     np       dist    gamma dir.hor dir.ver   id
1   454 0.08644121 15.24437       0       0 var1
2   922 0.31441297 38.01861       0       0 var1
3  1220 0.49499138 47.53232       0       0 var1
4  1599 0.71534068 59.90295       0       0 var1
5  1457 0.90005368 76.49265       0       0 var1
6  2231 1.09236560 78.85563       0       0 var1
------------------------------------------------------------
7  2264 1.30215002 89.44291       0       0 var1
8  2466 1.50010567 79.60835       0       0 var1
9  2256 1.70695699 89.64108       0       0 var1
10 2118 1.89091691 68.36358       0       0 var1
11 2256 2.09530679 77.57633       0       0 var1
12  176 2.21278828 83.79775       0       0 var1


-----

nugget=7.943094, range=1.276398, partial sill=74.333216,
sill=74.333216+7.943094= 82.27631

------

Classical linearized version:

rectacla2<-lm(nn$gamma[1:6]-7.943094~x[1:6]-1)
> rectacla2$coefficients
  x[1:6]
74.4912


7.943094 + 74.4912 * h    ,  h =< 74.333216/74.4912 =   0.9978792
82.27631    ,  h >   0.9978792
```

```
clasivarilinemodel2<-function(h)
{
  82.27631*ifelse(h>0.9978792,1,(7.943094 + 74.4912 * h) /82.27631)
}

-------

0.1-trimmed linearized version:


rectarecor2<-lm(m[1:6]-7.943094~x[1:6]-1)

> rectarecor2$coefficients
  x[1:6]
49.97665

> mean(m[7:12])
[1]  52.09326


7.943094+49.97665  * h   ,  h =< (52.09326-7.943094)/49.97665 = 0.8834159
   52.09326   ,  h > 0.8834159


recorvarilinemodel2<-function(h)
{
 52.09326*ifelse(h>0.8834159,1,(7.943094+49.97665  * h )/52.09326 )
}

----

Huber's linearized version:

rectahuber2<-lm(jt[1:6]-7.943094~x[1:6]-1)

> rectahuber2$coefficients
  x[1:6]
48.81407

> mean(jt[7:12])
[1] 54.58358


 7.943094+ 48.81407  * h   ,  h =< (54.58358-7.943094)/48.81407  = 0.9554722
54.58358    ,  h > 0.9554722

hubervarilinemodel2<-function(h)
{
  54.58358  *ifelse(h>0.9554722,1,(7.943094+ 48.81407  * h)/54.58358  )
}
```

Figure 8 of the paper is obtained with

```
x<-seq(0,12*0.2,len=12)
plot(x,nn$gamma,ylim=c(0,100),pch=16,ylab="semivariogram",xlab="h",main="Nickel")
points(x,m,pch=16,col=3)
points(x,jt,pch=16,col=2)
text(1.3,30,"     *Matheron")
text(1.3,25,"           *0.1-trimmed",col=3)
text(1.3,20,"*Huber",col=2)

h<-seq(0,12*0.2,len=10000)
lines(h,clasivarilinemodel2(h),type="l")
lines(h,recorvarilinemodel2(h),type="l",col=3)
lines(h,hubervarilinemodel2(h),type="l",col=2)

---

Global tests:

Classical,

x<-seq(0,12*0.2,len=12)


Statistics S_n:

> nn
     np       dist    gamma dir.hor dir.ver   id
1   454 0.08644121 15.24437       0       0 var1
2   922 0.31441297 38.01861       0       0 var1
3  1220 0.49499138 47.53232       0       0 var1
4  1599 0.71534068 59.90295       0       0 var1
5  1457 0.90005368 76.49265       0       0 var1
6  2231 1.09236560 78.85563       0       0 var1
7  2264 1.30215002 89.44291       0       0 var1
8  2466 1.50010567 79.60835       0       0 var1
9  2256 1.70695699 89.64108       0       0 var1
10 2118 1.89091691 68.36358       0       0 var1
11 2256 2.09530679 77.57633       0       0 var1
12  176 2.21278828 83.79775       0       0 var1

> 2*clasivarilinemodel2(x)
 [1]  15.88619  48.39144  80.89669 113.40194 145.90719 164.55262 164.55262
 [8] 164.55262 164.55262 164.55262 164.55262 164.55262

> max(abs(2*nn$gamma-2*clasivarilinemodel2(x)))
```

```
[1] 27.82546
```

1-p-value $= F_{S_n}(v)$:

```
(1-0)*
(approx(0.01,922,-27.82546+48.39144,48.39144,1.1)-approx(0.01,922,27.82546+48.39144,48.39144,1.1))*
(approx(0.01,1220,-27.82546+80.89669,80.89669,1.1)-approx(0.01,1220,27.82546+80.89669,80.89669,1.1))*
(approx(0.01,1599,-27.82546+113.40194,113.40194,1.1)-approx(0.01,1599,27.82546+113.40194,113.40194,1.1))*
(approx(0.01,1457,-27.82546+145.90719,145.90719,1.1)-approx(0.01,1457,27.82546+145.90719,145.90719,1.1))*
(approx(0.01,2231,-27.82546+164.55262,164.55262,1.1)-approx(0.01,2231,27.82546+164.55262,164.55262,1.1))*
(approx(0.01,2264,-27.82546+164.55262,164.55262,1.1)-approx(0.01,2264,27.82546+164.55262,164.55262,1.1))*
(approx(0.01,2466,-27.82546+164.55262,164.55262,1.1)-approx(0.01,2466,27.82546+164.55262,164.55262,1.1))*
(approx(0.01,2256,-27.82546+164.55262,164.55262,1.1)-approx(0.01,2256,27.82546+164.55262,164.55262,1.1))*
(approx(0.01,2118,-27.82546+164.55262,164.55262,1.1)-approx(0.01,2118,27.82546+164.55262,164.55262,1.1))*
(approx(0.01,2256,-27.82546+164.55262,164.55262,1.1)-approx(0.01,2256,27.82546+164.55262,164.55262,1.1))*
(approx(0.01,176,-27.82546+164.55262,164.55262,1.1)-approx(0.01,176,27.82546+164.55262,164.55262,1.1))
[1] 0.8879351
```

```
Hence, p-value = 1-0.8879351 = 0.1120649

---------

0.1-trimmed,

Statistics S_n:

> 2*recorvarilinemodel2(x)
 [1]  15.88619  37.69418  59.50217  81.31017 103.11816 104.18652 104.18652
 [8] 104.18652 104.18652 104.18652 104.18652 104.18652

> max(abs(2*m-2*recorvarilinemodel2(x)))
[1] 58.39081
```

1-p-value $= F_{S_n}(v)$:

```
alpha<-0.1
ite<-20000
c2<-(1/((1-2*alpha)^(1/(ite+1))))-1
c1<-((1-2*alpha)^(1/(ite+1)))-1
(((1+454*c1)^(ite+1))*((1+454*c2)^(ite+1))*(1-0))*
(((1+922*c1)^(ite+1))*((1+922*c2)^(ite+1))*(1-0))*
(((1+1220*c1)^(ite+1))*((1+1220*c2)^(ite+1))*(approx(0.01,1220,-58.39081+59.50217,59.50217,1.1)
-approx(0.01,1220,58.39081+59.50217,59.50217 ,1.1)))*
(((1+1599*c1)^(ite+1))*((1+1599*c2)^(ite+1))*(approx(0.01,1599,-58.39081+81.31017,81.31017,1.1)
-approx(0.01,1599,58.39081+81.31017,81.31017,1.1)))*
(((1+1457*c1)^(ite+1))*((1+1457*c2)^(ite+1))*(approx(0.01,1457,-58.39081+103.11816,103.11816,1.1)
-approx(0.01,1457,58.39081+103.11816,103.11816,1.1)))*
(((1+2231*c1)^(ite+1))*((1+2231*c2)^(ite+1))*(approx(0.01,2231,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,2231,58.39081+104.18652,104.18652,1.1)))*
(((1+2264*c1)^(ite+1))*((1+2264*c2)^(ite+1))*(approx(0.01,2264,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,2264,58.39081+104.18652,104.18652,1.1)))*
(((1+2466*c1)^(ite+1))*((1+2466*c2)^(ite+1))*(approx(0.01,2466,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,2466,58.39081+104.18652,104.18652,1.1)))*
(((1+2256*c1)^(ite+1))*((1+2256*c2)^(ite+1))*(approx(0.01,2256,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,2256,58.39081+104.18652,104.18652,1.1)))*
(((1+2118*c1)^(ite+1))*((1+2118*c2)^(ite+1))*(approx(0.01,2118,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,2118,58.39081+104.18652,104.18652,1.1)))*
(((1+2256*c1)^(ite+1))*((1+2256*c2)^(ite+1))*(approx(0.01,2256,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,2256,58.39081+104.18652,104.18652,1.1)))*
(((1+176*c1)^(ite+1))*((1+176*c2)^(ite+1))*(approx(0.01,176,-58.39081+104.18652,104.18652,1.1)
-approx(0.01,176,58.39081+104.18652,104.18652,1.1)))
[1] 6.474842e-42
```

```
p-value:

> 1-6.474842e-42
[1] 1


--------

Huber,


> 2*hubervarilinemodel2(x)
 [1]  15.88619  37.18687  58.48756  79.78824 101.08893 109.16716 109.16716
 [8] 109.16716 109.16716 109.16716 109.16716 109.16716

Statistics S_n:

> max(abs(2*jt-2*hubervarilinemodel2(x)))
[1] 29.09298
```

-----

library(MASS)

```
> saddle(0.5,15.88619,29.09298)
[1] -1.508397e-05

> saddle(0.5,37.18687,29.09298)
[1] -1.414718e-05

> saddle(0.5,58.48756,29.09298)
[1] -1.344658e-05

> saddle(0.5,79.78824,29.09298)
[1] -1.285565e-05

> saddle(0.5,101.08893,29.09298)
[1] -1.233127e-05

> saddle(0.5,109.16716,29.09298)
[1] -1.214442e-05
```

Hence, z0=0.5 is valid

VOM+SAD approximation (13):

```
approxhuber<-function(e,n,t,va,g,z0,b)
{
dentrog<-function(x){exp(z0*huber(x-t,1)$mu)*dgamma(x,0.5,2*(g^2)*va)}
Bg<-integrate(dentrog,-Inf,Inf)$value
dentro1<-function(x){exp(z0*huber(x-t,1)$mu)*dgamma(x,0.5,2*va)}
B1<-integrate(dentro1,-Inf,Inf)$value
s<-sqrt(-2*n*log(B1))
dentro2<-function(x){exp(z0*huber(x-t,1)$mu)*((huber(x-t,1)$mu)^2)*dgamma(x,0.5,2*va)}
B2<-integrate(dentro2,-Inf,Inf)$value
r1<-z0*sqrt(B2)
r<-sqrt(n)*r1
A<-1-pnorm(s)+dnorm(s)*(1/r+1/s)
resul<-A+e*sqrt(n)*dnorm(s)/r1*((Bg/B1)-1)
return(resul)
}
```

1-p-value:

```
library(MASS)
(1-0)*
(approxhuber(0.01,922,-29.09298+37.18687,37.18687,1.1,0.5,1)
-approxhuber(0.01,922,29.09298+37.18687,37.18687,1.1,0.5,1))*
(approxhuber(0.01,1220,-29.09298+58.48756,58.48756,1.1,0.5,1)
-approxhuber(0.01,1220,29.09298+58.48756,58.48756,1.1,0.5,1))*
(approxhuber(0.01,1599,-29.09298+79.78824,79.78824,1.1,0.5,1)
-approxhuber(0.01,1599,29.09298+79.78824,79.78824,1.1,0.5,1))*
(approxhuber(0.01,1457,-29.09298+101.08893,101.08893,1.1,0.5,1)
-approxhuber(0.01,1457,29.09298+101.08893,101.08893,1.1,0.5,1))*
(approxhuber(0.01,2231,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,2231,29.09298+109.16716,109.16716,1.1,0.5,1))*
(approxhuber(0.01,2264,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,2264,29.09298+109.16716,109.16716,1.1,0.5,1))*
(approxhuber(0.01,2466,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,2466,29.09298+109.16716,109.16716,1.1,0.5,1))*
(approxhuber(0.01,2256,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,2256,29.09298+109.16716,109.16716,1.1,0.5,1))*
(approxhuber(0.01,2118,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,2118,29.09298+109.16716,109.16716,1.1,0.5,1))*
(approxhuber(0.01,2256,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,2256,29.09298+109.16716,109.16716,1.1,0.5,1))*
(approxhuber(0.01,176,-29.09298+109.16716,109.16716,1.1,0.5,1)
-approxhuber(0.01,176,29.09298+109.16716,109.16716,1.1,0.5,1))
[1] 0
```

Then, p-value = 1.

Example 3

Let us consider the geolocated *pollution data*

```
Location_number Station_location Population NO NO2 PM10 O3 xPol yPol
1 Alcala_de_Henares 194310 23.6 37.0 26.5 53.3 -3.377949 40.479328
2 Alcobendas 114864 17.1 32.3 21.1 57.4 -3.646455 40.539523
3 Aranjuez 58213 5.1 16.4 22.2 60.4 -3.591644444 40.03327778
4 Arganda_del_Rey 53821 9.0 24.0 24.1 52.7 -3.458830556 40.30069444
5 El_Atazar 97 1.1 5.2 14.6 85.2 -3.467902778 40.90901944
6 Colmenar_Viejo 48614 10.1 27.3 19.7 62.0 -3.773865 40.664649
7 Coslada 83011 31.6 47.2 26.7 45.0 -3.542261 40.430461
8 Fuenlabrada 194669 15.3 36.5 21.6 53.8 -3.800946 40.281505
9 Getafe 178288 29.4 42.5 25.5 50.4 -3.716868 40.314518
10 Guadalix_de_la_Sierra 6049 3.6 12.4 19.1 65.3 -3.702147222 40.7806333
11 Leganes 187720 29.4 43.1 25.4 46.2 -3.754508 40.339762
12 Majadahonda 71299 9.6 30.2 17.3 56.3 -3.868994444 40.44610278
13 Mostoles 206589 13.7 32.2 21.5 51.6 -3.876772 40.324225
14 Orusco_de_Tajua 1218 1.1 5.4 16.8 81.4 -3.221094444 40.28755556
15 Rivas_Vaciamadrid 83767 22.4 38.5 22.9 52.0 -3.542902778 40.35970556
16 San_Martin_de_Valdeiglesias 8298 2.1 10.0 19.8 65.4 -4.398116667 40.36775833
17 Torrejon_de_Ardoz 128013 12.8 30.7 25.4 51.4 -3.477645 40.449541
18 Villa_del_Prado 6337 1.5 13.5 21.5 63.7 -4.275671 40.248730
19 Madrid_(Aguirre) 3182981 33.6 62.9 19.3 41.4 -3.6823158 40.4215533
20 Madrid_(Farolillo) 3182981 22.1 42.4 24.3 46.5 -3.7318356 40.3947825
21 Madrid_(Casa_de_Campo) 3182981 9.7 25.5 20.0 58.3 -3.7473445 40.4193577
22 Madrid_(Tres_Olivos) 3182981 14.2 36.1 20.1 57.2 -3.6897308 40.5005477
```

Two of these 4 variables are strongly correlated and have a distribution similar to a scale contaminated normal model; they are `NO` and `NO2`. We see this with the following sentences:

```
par(mfrow=c(1,2))
qqnorm(pollution$NO)
qqnorm(pollution$NO2)

cor(pollution$NO,pollution$NO2)
```

Because the coordinates in the pollution data set are `xPol` (longitude) and `yPol` (latitude), we establish these with the sentences

```
library(gstat)
library(sp)
coordinates(pollution) = ~xPol+yPol
```

Then, we create a *gstat* object with

```
g <- gstat(NULL, "NO", NO ~ 1, pollution)
g <- gstat(g, "NO2", NO2 ~ 1, pollution)
```

We compute, and plot, the variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations, with

```
vvm<-variogram(g,width=0.05)
plot(vvm)
```

obtaining Fig. 12. From this cross-variogram plot we can admit, initially, a Matern model, given in Michael Steins book, with sill=psill=150, range=0.3 and nugget=0, model that we establish with

```
model<-vgm(150,"Ste",0.3,0)
```

Now we check the adequacy between the model and the classical cross-variogram estimator with

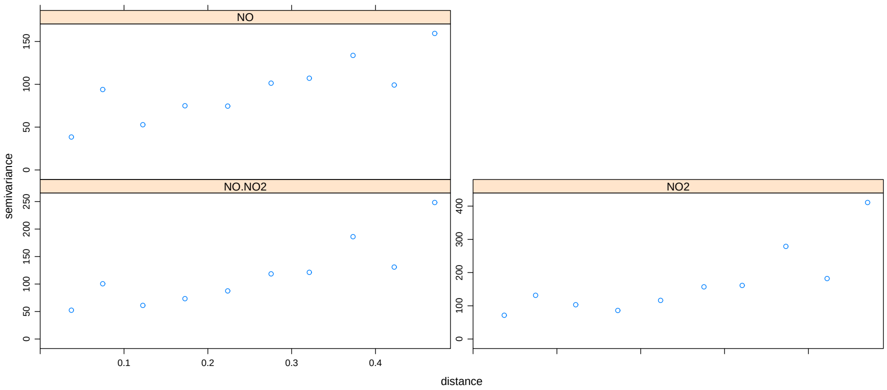

Figure 12: Variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations of Example 3.

```
> fit.lmc(vvm,g,model)
data:
NO : formula = NO`~`1 ; data dim = 22 x 7
NO2 : formula = NO2`~`1 ; data dim = 22 x 7
variograms:
          model     psill range kappa
NO[1]       Nug  47.31086   0.0   0.0
NO[2]       Ste  67.93841   0.3   0.5
NO2[1]      Nug  65.17751   0.0   0.0
NO2[2]      Ste 126.69361   0.3   0.5
NO.NO2[1]   Nug  50.82919   0.0   0.0
NO.NO2[2]   Ste  85.05376   0.3   0.5
```

From this result we shall consider a Matern model (`Ste`) for the variograms a the cross-variogram. For this last one, from the last two lines we obtain a better model, that should be a Matern model with partial sill=85.05376, range=0.3, nugget=50.82919 and kappa=0.5, model that we establish now

```
model2<-vgm(85.05376,"Ste",0.3,50.82919,fit.kappa=0.5)
```

We obtain a picture of the data and the best model in Fig. 13, with the sentence

```
plot(vvm,fit.lmc(vvm,g,model2))
```

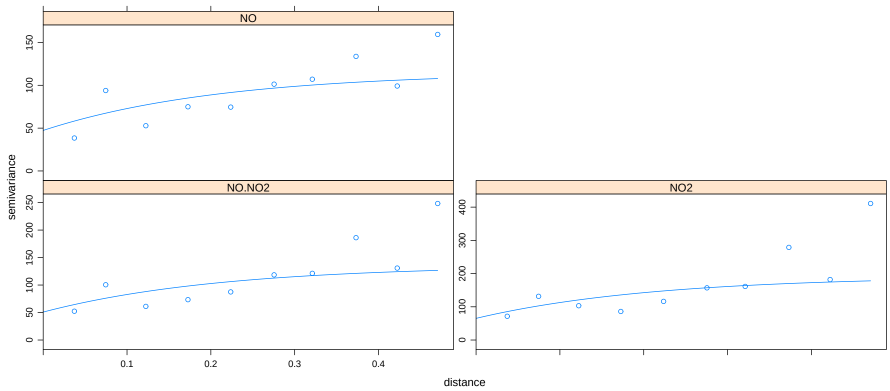

Figure 13: Variogram-crossvariogram matrix of the classical variogram and cross-variogram estimations with the classical model of Example 3.

To extract from `vvm` the values of the classical cross-variogram estimator, plotted in the previous picture, we define

```
vvmcross<-vvm[vvm$id == "NO.NO2",]
```

To compute the classical estimations (again) and the robust ones, we define

```
vm<-variogram(g,width=0.05,cloud=T)
vmcross<-vm[vm$id == "NO.NO2",]
```

The previous classical estimations of the cross-variogram are obtained with

```
vvmcross$gamma
```

or with

```
mean(vmcross[vmcross$dist < 0.05,]$gamma)
mean(vmcross[vmcross$dist > 0.05 & vmcross$dist<0.10,]$gamma)
mean(vmcross[vmcross$dist > 0.10 & vmcross$dist<0.15,]$gamma)
mean(vmcross[vmcross$dist > 0.15 & vmcross$dist<0.20,]$gamma)
mean(vmcross[vmcross$dist > 0.20 & vmcross$dist<0.25,]$gamma)
mean(vmcross[vmcross$dist > 0.25 & vmcross$dist<0.30,]$gamma)
mean(vmcross[vmcross$dist > 0.30 & vmcross$dist<0.35,]$gamma)
mean(vmcross[vmcross$dist > 0.35 & vmcross$dist<0.40,]$gamma)
```

```
mean(vmcross[vmcross$dist > 0.40 & vmcross$dist<0.45,]$gamma)
mean(vmcross[vmcross$dist > 0.45,]$gamma)
```

The 0.1-trimmed variogram estimations are computed with

```
y1<-mean(vmcross[vmcross$dist < 0.05,]$gamma,0.1)
y2<-mean(vmcross[vmcross$dist > 0.05 & vmcross$dist<0.10,]$gamma,0.1)
y3<-mean(vmcross[vmcross$dist > 0.10 & vmcross$dist<0.15,]$gamma,0.1)
y4<-mean(vmcross[vmcross$dist > 0.15 & vmcross$dist<0.20,]$gamma,0.1)
y5<-mean(vmcross[vmcross$dist > 0.20 & vmcross$dist<0.25,]$gamma,0.1)
y6<-mean(vmcross[vmcross$dist > 0.25 & vmcross$dist<0.30,]$gamma,0.1)
y7<-mean(vmcross[vmcross$dist > 0.30 & vmcross$dist<0.35,]$gamma,0.1)
y8<-mean(vmcross[vmcross$dist > 0.35 & vmcross$dist<0.40,]$gamma,0.1)
y9<-mean(vmcross[vmcross$dist > 0.40 & vmcross$dist<0.45,]$gamma,0.1)
y10<-mean(vmcross[vmcross$dist > 0.45,]$gamma,0.1)

y<-c(y1,y2,y3,y4,y5,y6,y7,y8,y9,y10)
```

The Huber's cross-variogram estimations are computed with

```
library(MASS)

u1<-huber(vmcross[vmcross$dist < 0.05,]$gamma,0.1)$mu
u2<-huber(vmcross[vmcross$dist > 0.05 & vmcross$dist<0.10,]$gamma)$mu
u3<-huber(vmcross[vmcross$dist > 0.10 & vmcross$dist<0.15,]$gamma)$mu
u4<-huber(vmcross[vmcross$dist > 0.15 & vmcross$dist<0.20,]$gamma)$mu
u5<-huber(vmcross[vmcross$dist > 0.20 & vmcross$dist<0.25,]$gamma)$mu
u6<-huber(vmcross[vmcross$dist > 0.25 & vmcross$dist<0.30,]$gamma)$mu
u7<-huber(vmcross[vmcross$dist > 0.30 & vmcross$dist<0.35,]$gamma)$mu
u8<-huber(vmcross[vmcross$dist > 0.35 & vmcross$dist<0.40,]$gamma)$mu
u9<-huber(vmcross[vmcross$dist > 0.40 & vmcross$dist<0.45,]$gamma)$mu
u10<-huber(vmcross[vmcross$dist > 0.45,]$gamma)$mu

u<-c(u1,u2,u3,u4,u5,u6,u7,u8,u9,u10)
```

The classical, 0.1-trimmed and Huber's estimations are

```
> vvmcross$gamma
 [1]  52.39000 100.52115  61.15500  73.40423  87.49452 118.51294 121.30158
 [8] 186.15475 130.86406 248.32708

> y
 [1]  52.39000  78.66864  49.26846  53.95643  67.56647  94.46964 112.10672
 [8] 144.13875 114.31577 204.37350
```

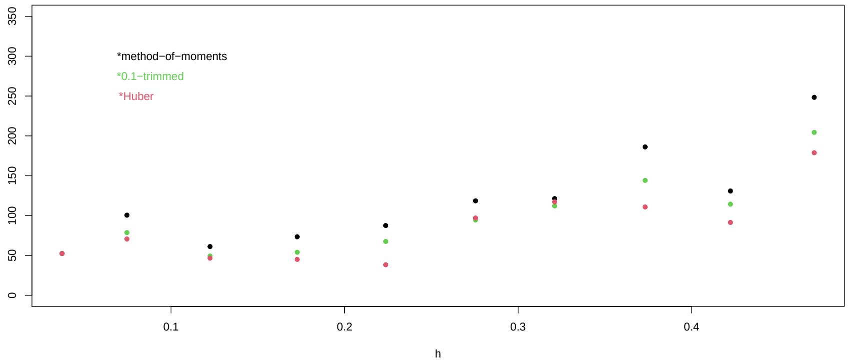

Figure 14: Classical (black) and robust (green and red) cross-variogram estimations of Example 3.

```
> u
 [1]  52.39000  70.63021  46.59529  45.01145  38.30045  97.00027 117.14915
 [8] 110.82071  91.40090 178.83960
```

These estimations are plotted in Fig. 14
obtained with

```
plot(vvmcross$dist,vvmcross$gamma,ylim=c(0,350),pch=16,ylab=" ",xlab="h")
points(vvmcross$dist,y,pch=16,col=3)
points(vvmcross$dist,u,pch=16,col=2)
text(0.08,300,"                        *method-of-moments")
text(0.08,275,"         *0.1-trimmed",col=3)
text(0.08,250,"*Huber",col=2)
```

Because of the shape of the Matern model, that we see in Fig. 13, and because the marginal fit models give a range larger that 0.5, we shall use as linearized versions of the semi-variograms and semi-cross-variogram, the classical regression line of the pairs $(||\mathbf{h}||, \hat{\gamma}(\mathbf{h})),\ \forall\, ||\mathbf{h}||$. For the cross-variogram, the classical, 0.1-trimmed and Huber's are, respectively

```
recta0<-lm(vvmcross$gamma~vvmcross$dist)
recta1<-lm(y~vvmcross$dist)
recta2<-lm(u~vvmcross$dist)
```

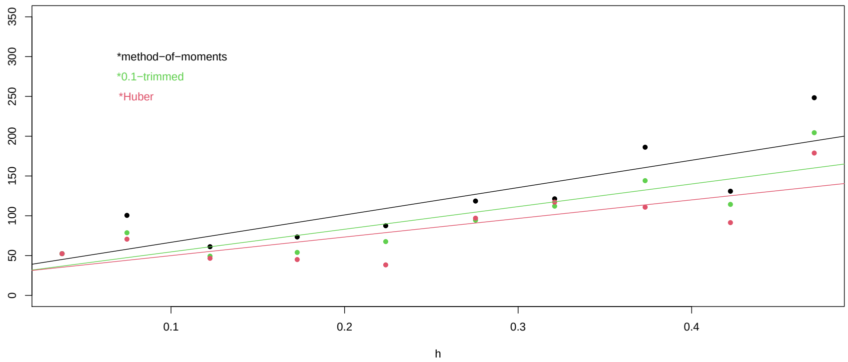

Figure 15: Classical (black) and robust (green and red) cross-variogram estimations of Example 3, with the linearized cross-variogram models.

all of them significant and that can be used for cokriging.

Nevertheless, we can appreciate the influence of the outliers in the estimation of the (linearized) cross-variogram in Fig. 15 (which is Figure 11 of the paper) obtained with the following sentences:

```
plot(vvmcross$dist,vvmcross$gamma,ylim=c(0,350),pch=16,ylab=" ",xlab="h")
points(vvmcross$dist,y,pch=16,col=3)
points(vvmcross$dist,u,pch=16,col=2)
abline(recta0)
abline(recta1,col=3)
abline(recta2,col=2)
text(0.08,300,"                        *method-of-moments")
text(0.08,275,"         *0.1-trimmed",col=3)
text(0.08,250,"*Huber",col=2)
```