Assignment 2

2023-11-04

library(geoR)

## --------------------------------------------------------------  
## Analysis of Geostatistical Data  
## For an Introduction to geoR go to http://www.leg.ufpr.br/geoR  
## geoR version 1.9-2 (built on 2022-08-09) is now loaded  
## --------------------------------------------------------------

library(fields)

## Loading required package: spam

## Spam version 2.10-0 (2023-10-23) is loaded.  
## Type 'help( Spam)' or 'demo( spam)' for a short introduction   
## and overview of this package.  
## Help for individual functions is also obtained by adding the  
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

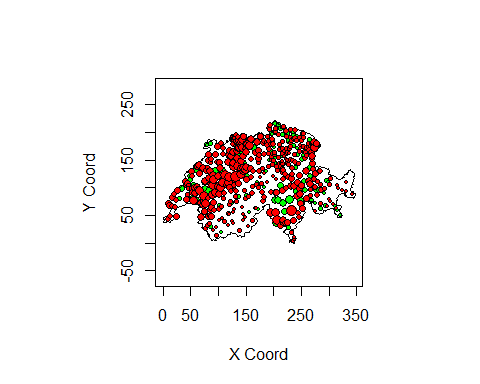
##   
## Attaching package: 'spam'

## The following objects are masked from 'package:base':  
##   
## backsolve, forwardsolve

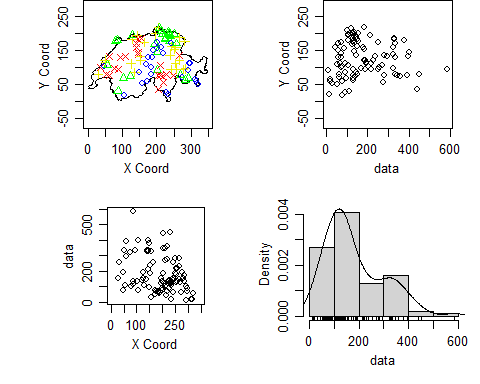
## Loading required package: viridisLite

##   
## Try help(fields) to get started.

points(sic.100, borders=sic.borders,col="green")  
points(sic.367, borders=sic.borders,col="red",add=TRUE)



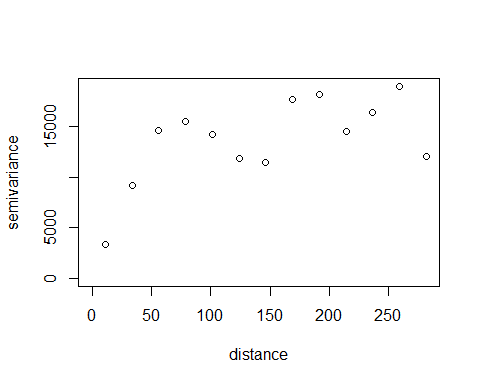
plot.geodata(sic.100,bor=sic.borders)



vario.b<- variog(sic.100,option =c ("bin", "cloud", "smooth"),  
bin.cloud=TRUE)

## variog: computing omnidirectional variogram

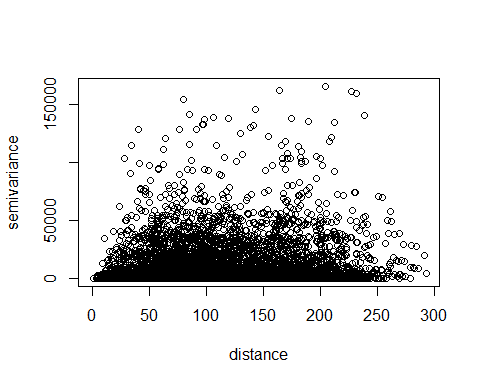
plot(vario.b)



vario.c <- variog(sic.100, op="cloud")

## variog: computing omnidirectional variogram

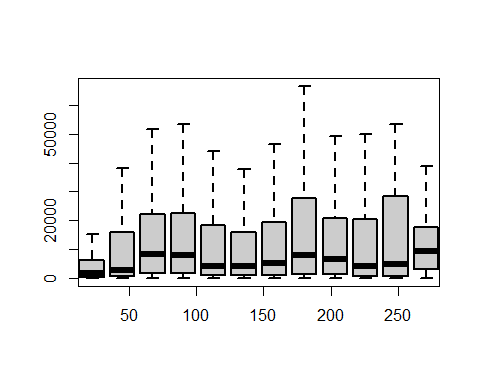
plot(vario.c)



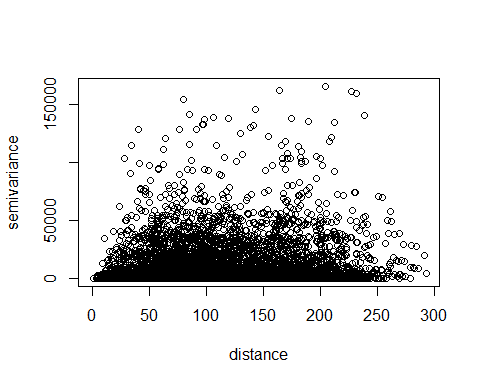
vario.c <- variog(sic.100, op="cloud")

## variog: computing omnidirectional variogram

bplot.xy(vario.c$u,vario.c$v, breaks=vario.b$u,col="grey80",  
lwd=2,cex=0.1,outline=FALSE)



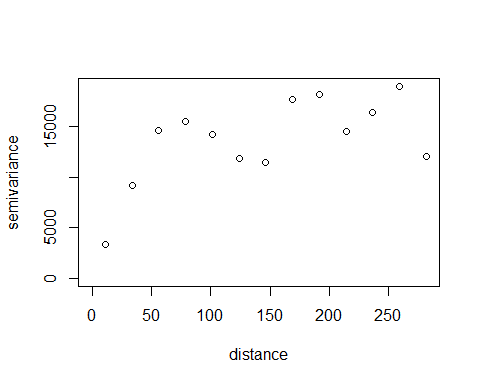
plot(vario.c)



vario.ex<- variog(sic.100, bin.cloud=TRUE)

## variog: computing omnidirectional variogram

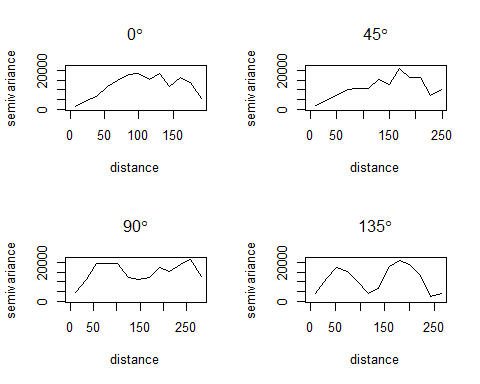
plot(vario.ex)



vario4<-variog4(sic.100)

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

plot(vario4,same=FALSE)



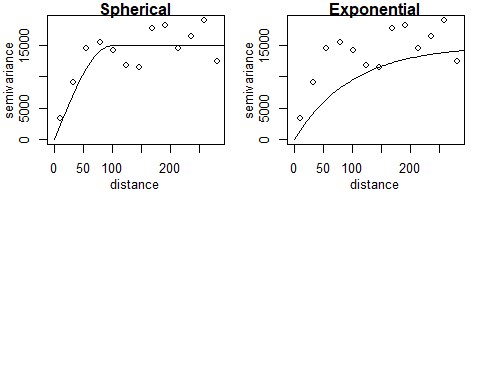
vario.ex<- variog(sic.100,option="bin")

## variog: computing omnidirectional variogram

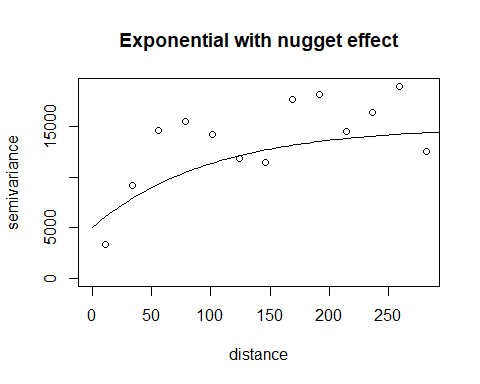
vario.sphe<-(variofit(vario.ex,cov.model= "spher",  
ini.cov.pars=c(15000,200)))

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

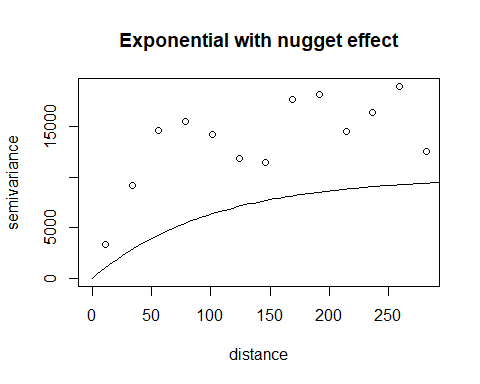
par(mfrow=c(2,2), mar=c(3,3,1,1), mgp =c (2,1,0))  
plot(vario.ex,main="Spherical")  
lines.variomodel(cov.model="sphe",cov.pars=c(15000,100),  
nug=0,max.dist=350)  
plot(vario.ex,main="Exponential")  
lines.variomodel(cov.model="exp",cov.pars=c(15000,100),  
nug=0,max.dist=350)



plot(vario.ex,main="Exponential with nugget effect")  
lines.variomodel(cov.model="exp",cov.pars=c(10000,100),  
nug=5000,max.dist=350)



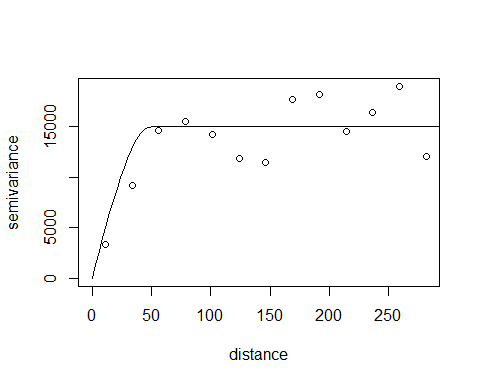
plot(vario.ex,main="Exponential with nugget effect")  
lines.variomodel(cov.model="matern",cov.pars=c(10000,100),  
nug=0,max.dist=350,kappa=0.5)



vario.ex<- variog(sic.100, bin.cloud=TRUE)

## variog: computing omnidirectional variogram

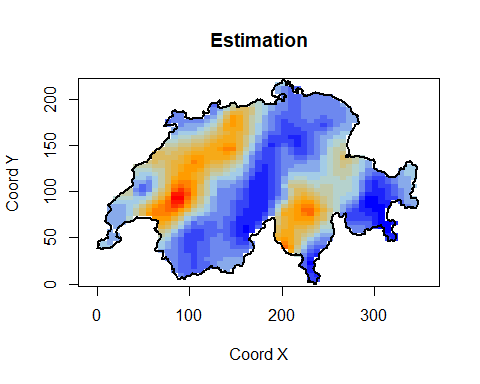
plot(vario.ex,main="")  
lines.variomodel(cov.model="spher",cov.pars=c(15000,50),  
nug=0,max.dist=300)



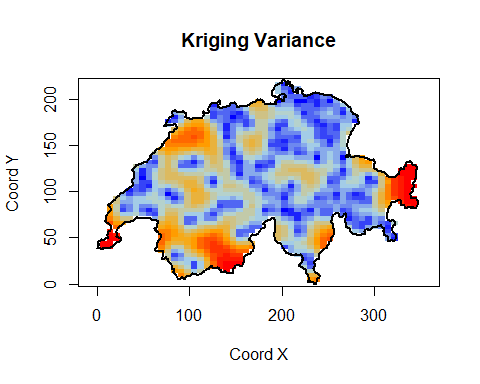
library(geoR)  
pred.grid <- expand.grid(seq(0, 350, length.out = 51), seq(0, 220, length.out = 51))  
rgb.palette <- colorRampPalette(c("blue", "lightblue", "orange", "red"), space = "rgb")  
kc <- krige.conv(sic.100, loc = pred.grid,  
 krige = krige.control(cov.model = "spherical", cov.pars = c(15000, 50)))

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

image(kc, loc = pred.grid, col = rgb.palette(20), xlab = "Coord X",  
 ylab = "Coord Y", borders = sic.borders, main = "Estimation")



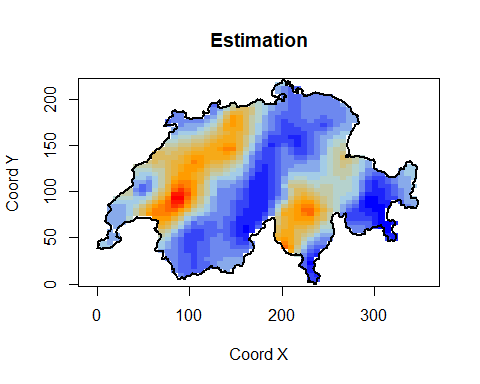
krige.var <- kc$krige.var  
image(kc, krige.var, loc = pred.grid, col = rgb.palette(20),  
 xlab = "Coord X", ylab = "Coord Y", borders = sic.borders,  
 main = "Kriging Variance")



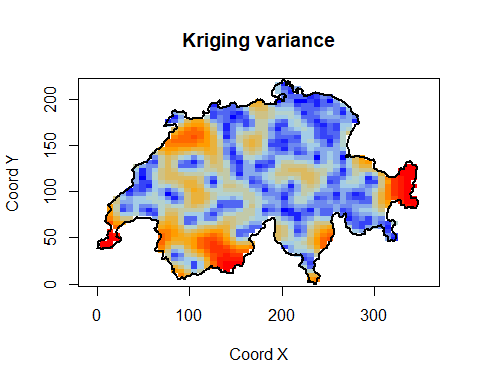
pred.grid <- expand.grid(seq(0,350, l=51),seq (0,220, l=51))  
rgb.palette <- colorRampPalette(c("blue", "lightblue",  
"orange", "red"),space = "rgb")  
kc<- krige.conv(sic.100, loc = pred.grid,  
krige=krige.control(cov.model="spherical",cov.pars=c(15000,50)))

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

image(kc, loc = pred.grid,col =rgb.palette(20) ,xlab="Coord X",  
ylab="Coord Y",borders=sic.borders,main="Estimation")



image(kc, krige.var,loc = pred.grid,col=rgb.palette(20),  
xlab="Coord X",ylab="Coord Y",borders=sic.borders,  
main="Kriging variance")



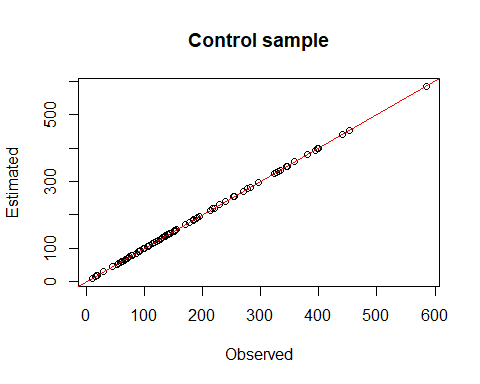
kc1<- krige.conv(sic.100, loc = sic.100$coords,  
krige=krige.control(cov.model="spherical",cov.pars=c(16000,47)))

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

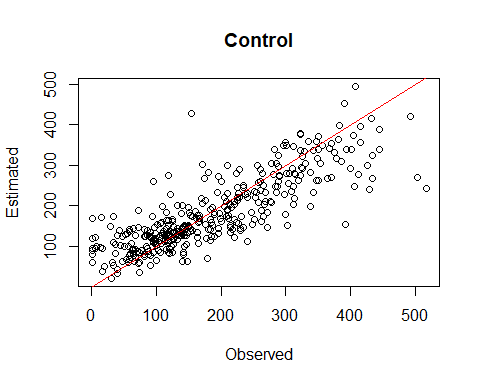
kc2<- krige.conv(sic.100, loc = sic.367$coords,  
krige=krige.control(cov.model="spherical",cov.pars=c(16000,47)))

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

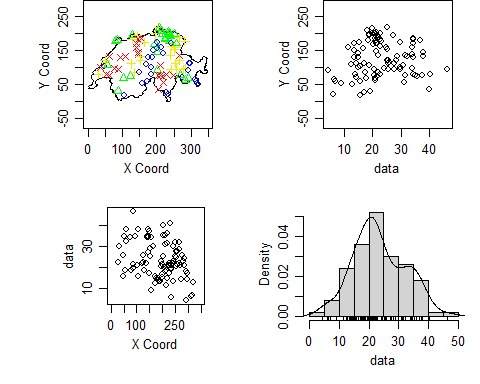
plot(sic.100$data,kc1$predict,xlab="Observed",ylab="Estimated",main="Control sample")  
abline(a=0,b=1,col="red")



plot(sic.367$data,kc2$predict,,xlab="Observed",ylab="Estimated",  
main="Control")  
abline(a=0,b=1,col="red")



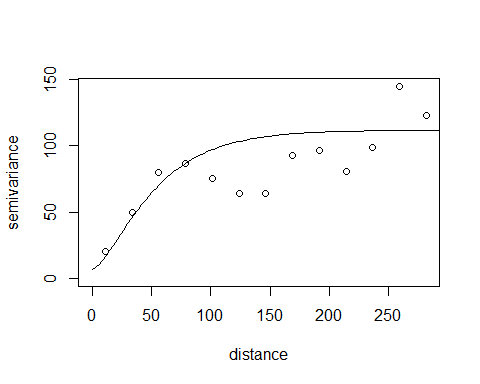
plot.geodata(sic.100,bor=sic.borders,lambda=0.5)



vario.ext<- variog(sic.100,option="bin",lambda=0.5)

## variog: computing omnidirectional variogram

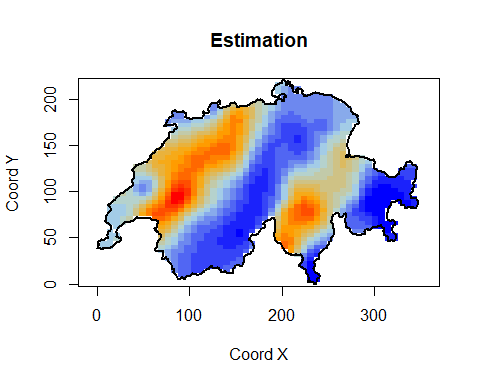
plot(vario.ext)  
lines.variomodel(cov.m = "mat",cov.p =c (105, 36), nug = 6.9,  
max.dist = 300,kappa = 1, lty = 1)



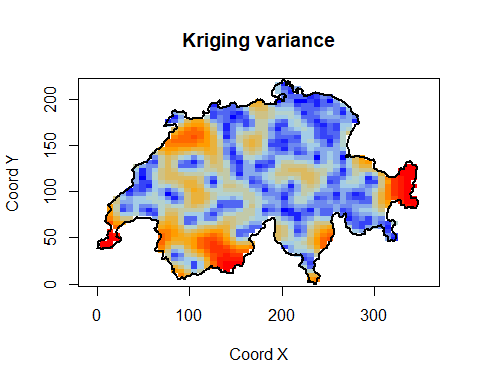
kct<- krige.conv(sic.100, loc = pred.grid,  
krige=krige.control(cov.model="matern",cov.pars=c(105, 36),  
kappa=1,nugget=6.9,lambda=0.5))

## krige.conv: model with constant mean  
## krige.conv: performing the Box-Cox data transformation  
## krige.conv: back-transforming the predicted mean and variance  
## krige.conv: Kriging performed using global neighbourhood

pred.grid <- expand.grid(seq(0,350, l=51),seq (0,220, l=51))  
rgb.palette <- colorRampPalette(c("blue", "lightblue",  
"orange", "red"),space = "rgb")  
image(kct, loc = pred.grid,col =rgb.palette(20) , xlab="Coord X",  
ylab="Coord Y",borders=sic.borders,main="Estimation")



image(kct, krige.var,loc = pred.grid,col =rgb.palette(20) ,  
xlab="Coord X",ylab="Coord Y",borders=sic.borders,  
main="Kriging variance")



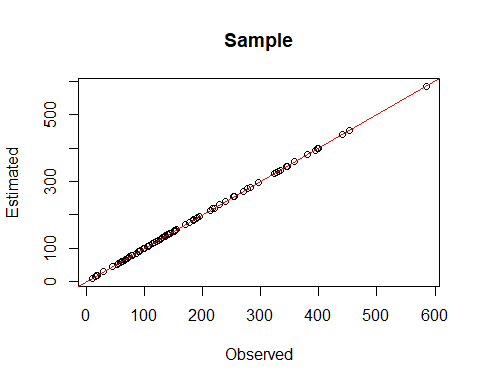
kct1<- krige.conv(sic.100, loc = sic.100$coords,  
krige=krige.control(cov.model="spherical",cov.pars=c(16000,47),  
kappa=1,nugget=6.9,lambda=0.5))

## krige.conv: model with constant mean  
## krige.conv: performing the Box-Cox data transformation  
## krige.conv: back-transforming the predicted mean and variance  
## krige.conv: Kriging performed using global neighbourhood

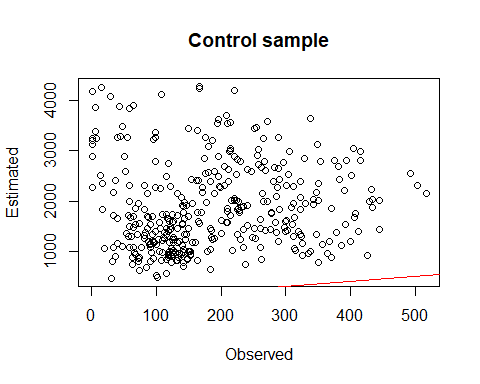
kct2<- krige.conv(sic.100, loc = sic.367$coords,  
krige=krige.control(cov.model="spherical",cov.pars=c(16000,47),  
kappa=1,nugget=6.9,lambda=0.5))

## krige.conv: model with constant mean  
## krige.conv: performing the Box-Cox data transformation  
## krige.conv: back-transforming the predicted mean and variance  
## krige.conv: Kriging performed using global neighbourhood

plot(sic.100$data,kct1$predict,xlab="Observed",ylab="Estimated",  
main="Sample")  
abline(a=0,b=1,col="red")



plot(sic.367$data,kct2$predict,,xlab="Observed",ylab="Estimated",  
main="Control sample")  
abline(a=0,b=1,col="red")



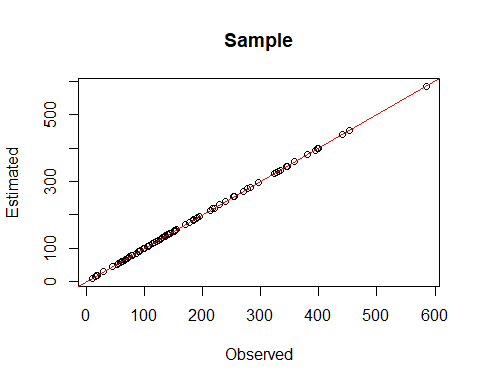
kct1 <- krige.conv(sic.100, loc = sic.100$coords,  
 krige = krige.control(cov.model = "spherical", cov.pars = c(16000, 47),  
 kappa = 1, nugget = 6.9, lambda = 0.5))

## krige.conv: model with constant mean  
## krige.conv: performing the Box-Cox data transformation  
## krige.conv: back-transforming the predicted mean and variance  
## krige.conv: Kriging performed using global neighbourhood

kct2 <- krige.conv(sic.100, loc = sic.367$coords,  
 krige = krige.control(cov.model = "spherical", cov.pars = c(16000, 47),  
 kappa = 1, nugget = 6.9, lambda = 0.5))

## krige.conv: model with constant mean  
## krige.conv: performing the Box-Cox data transformation  
## krige.conv: back-transforming the predicted mean and variance  
## krige.conv: Kriging performed using global neighbourhood

# Scale the y-axis values in the control sample estimated values by 0.01 (to show in hundreds)  
scaled\_kct2\_predict <- kct2$predict \* 0.1  
  
plot(sic.100$data, kct1$predict, xlab = "Observed", ylab = "Estimated", main = "Sample")  
abline(a = 0, b = 1, col = "red")



# For the control sample, use the scaled values  
plot(sic.367$data, scaled\_kct2\_predict, xlab = "Observed", ylab = "Estimated (in hundreds)", main = "Control sample")  
abline(a = 0, b = 1, col = "red")

