University of California, Los Angeles
Department of Statistics

Statistics C173/C273 **Instructor: Nicolas Christou**

Cross Validation

Cross validation is a technique that allows us to compare predicted values with true values. In spatial data this technique can help us to decide which variogram model to choose or which prediction method gives better results. The basic idea of cross validation is the following: We omit point i from the data set and we predict it using the remaining $n-1$ data points. Therefore, we can compare the predicted value with the true value at location s_i. Another way is to split the data set into two parts. The first part will be used for modeling the variogram. The spatial locations of the other part of the data set will be our grid. Once we predict the values we can compare them with the observed values at those locations.
We will use again the Maas river data from:

```
a <- read.table("http://www.stat.ucla.edu/~nchristo/
     statistics_c173_c273/soil.txt", header=T)
# Save the original image function:
image.orig <- image
```

We will randomly split the data in two parts. From the data, 100 observations will be used for modeling and 55 for prediction. Here are the commands:

```
choose100 <- sample(1:155, 100)
part_model <- a[choose100, ]
part_valid  <- a[-choose100, ]
```

Note: This is a random selection and every time we run these commands we will get different samples.

Now, we can use the `part_model` to estimate the variogram.

```
g <- gstat(id="log_lead", formula = log(lead)~1, locations = ~x+y,
data=part_model)

q <- variogram(g)
plot(q)

v.fit <- fit.variogram(q, vgm(1, "Sph", 800, 1))
plot(q, v.fit)
```

The predictions will be performed on the 55 spatial locations of the `part_valid` data set:

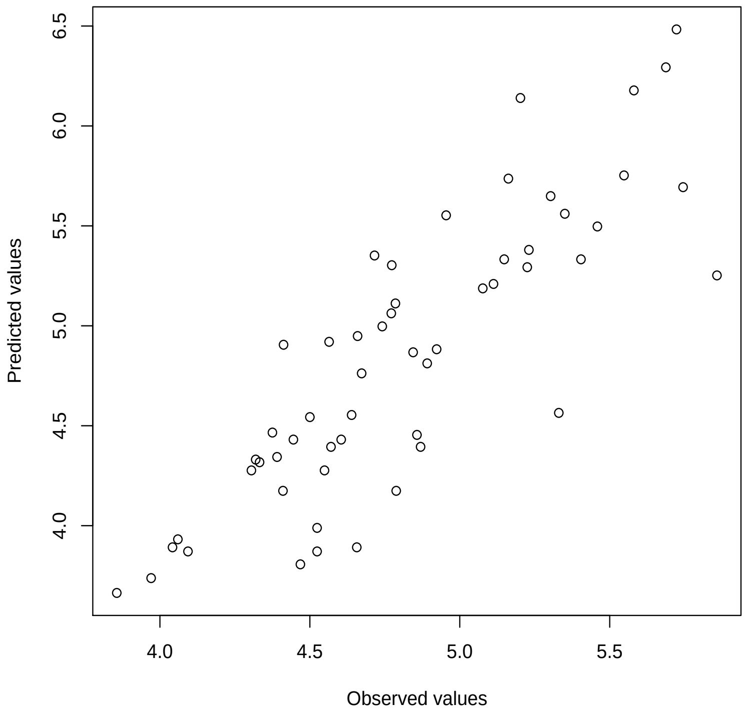
```
part_valid_pr <- krige(id="log_lead", log(lead)~1, locations=~x+y,
model=v.fit, data=part_model, newdata=part_valid)
```

Let's compute the difference between the predicted values and the true values:

```
difference <- log(part_valid$lead) - part_valid_pr$log_lead.pred
summary(difference)
```

A simple plot of the predicted values against the true values is shown below:

```
plot(part_valid_pr$log_lead.pred,log(part_valid$lead),
xlab="Observed values", ylab="Predicted values")
```



The correlation coefficient between these two sets of values is

```
> cor(part_valid_pr$log_lead.pred,log(part_valid$lead))
[1] 0.84808
```

And the corresponding coefficient of determination is $R^2 = \sqrt{0.83808} = 0.71924$. Obviously, with this value of R^2 the kriging prediction are better estimates than the sample mean.

A more automated way is to use the function `krige.cv`:

```
cv_pr <- krige.cv(log(lead)~1, data=a, locations=~x+y,
model=v.fit, nfold=nrow(a))
```

With `nfold=nrow(a)` we remove one data value at a time and we predict it from the remaining $n-1$ data.
The `krige.cv` function returns the following:

```
> names(cv_pr)
[1] "var1.pred" "var1.var"  "observed"  "residual"  "zscore"
[6] "fold"      "x"         "y"
> summary(cv_pr)
```

```
   var1.pred          var1.var          observed          residual
 Min.   :3.753   Min.   :0.1400   Min.   :3.611   Min.   :-1.0090977
 1st Qu.:4.380   1st Qu.:0.1690   1st Qu.:4.284   1st Qu.:-0.2250461
 Median :4.799   Median :0.1835   Median :4.812   Median :-0.0192036
 Mean   :4.808   Mean   :0.1908   Mean   :4.807   Mean   :-0.0007497
 3rd Qu.:5.191   3rd Qu.:0.2022   3rd Qu.:5.333   3rd Qu.: 0.1935486
 Max.   :6.030   Max.   :0.4481   Max.   :6.483   Max.   : 1.6266221
     zscore                 fold              x                 y
 Min.   :-2.3277046   Min.   :  1.0   Min.   :178605   Min.   :329714
 1st Qu.:-0.5095453   1st Qu.: 39.5   1st Qu.:179371   1st Qu.:330762
 Median :-0.0407317   Median : 78.0   Median :179991   Median :331633
 Mean   :-0.0008231   Mean   : 78.0   Mean   :180005   Mean   :331635
 3rd Qu.: 0.4689051   3rd Qu.:116.5   3rd Qu.:180630   3rd Qu.:332463
 Max.   : 3.7613572   Max.   :155.0   Max.   :181390   Max.   :333611
```

The `zscore` values are computed with the following formula:

$$z_i = \frac{z(s_i) - \hat{z}_{[i]}(s_i)}{\sigma_{[i]}(s_i)}$$

where, z_i is the z-score
$z(s_i)$ is the observed value of the omitted point
$\hat{z}_{[i]}(s_i)$ is the predicted value of the omitted point
$\sigma_{[i]}(s_i)$ is the corresponding kriging standard error.

The z-scores should have mean close to zero and variance close to 1.

If we want to compare two variograms (exponential and spherical), or choose among prediction methods (e.g. ordinary kriging, univeral kriging, inverse distance weighted interpolation method, etc.), or between different types of variograms (classical or robust), or different weights, we can use the prediction sum of squares. In general we will select the method with the smallest `PRESS`.

$$\text{PRESS} = \sum_{i=1}^{n} \left(z(s_i) - \hat{z}_{[i]}(s_i) \right)^2$$

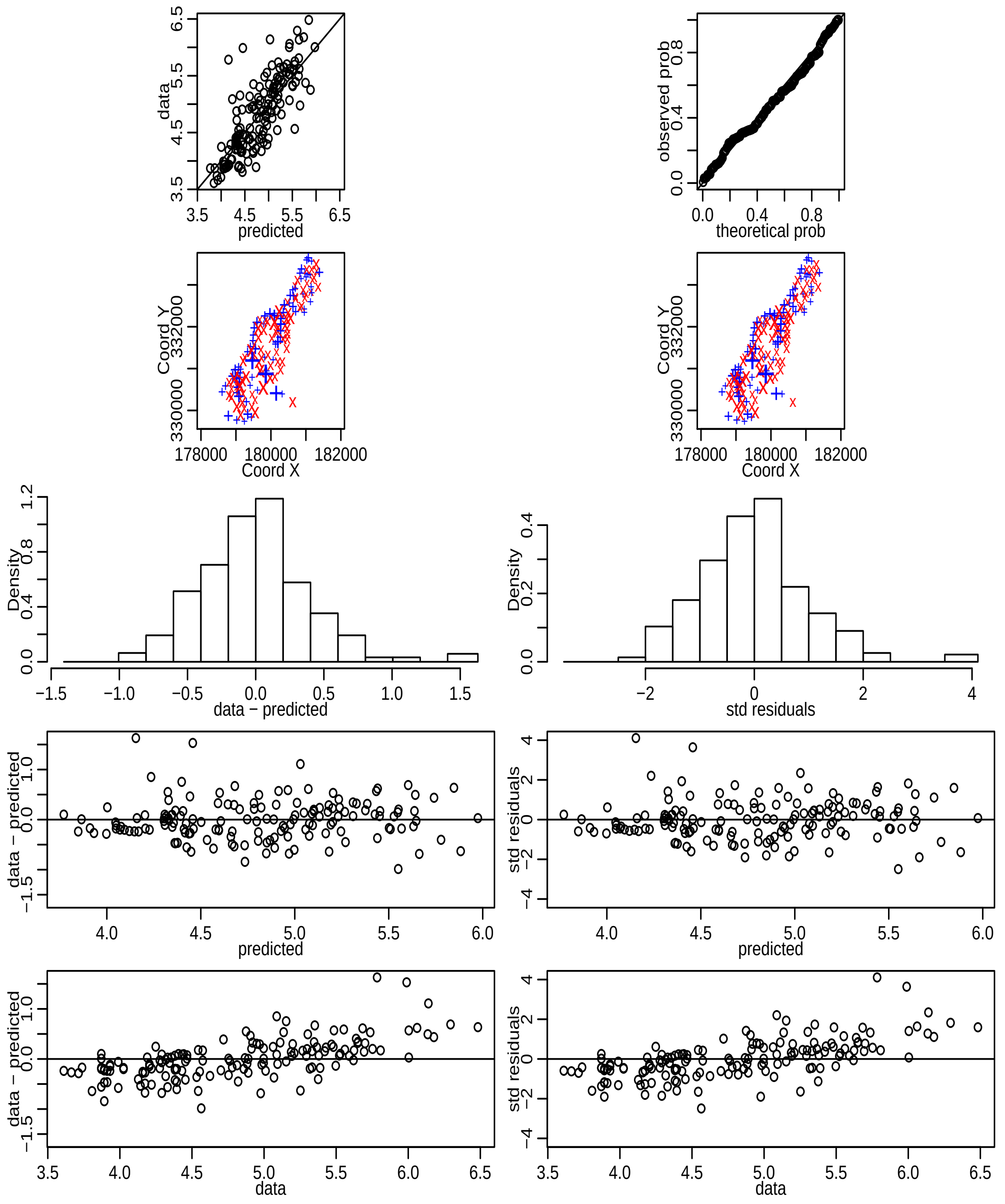
Cross validation using geoR:
Cross validation in `geoR` can be performed with the function `xvalid` using the strategy that omits one point at a time. The user can also provide other locations to be cross validated using the argument `location.xvalid`. Here is an example:

```
#Construct a data frame with x, y, and log(lead):
bb <- cbind(a$x, a$y, log(a$lead))

#Convert the data frame into a geodata object:
bbb <- as.geodata(bb)

#Computes the variogram with direction SW to NE:
var1 <- variog(bbb, dir=pi/4)

#Plot the variogram:
plot(var1)

#Fit the spherical variogram to the sample variogram:
fit1 <- variofit(var1, cov.model="sph", ini.cov.pars=c(0.3,1000),
fix.nugget=FALSE, nugget=0.1, wei="cressie")

#Draw the model variogram:
lines(fit1)

#Perform cross validation:
x_val1 <- xvalid(bbb, model=fit1)
```

We obtain several plots:

```
par(mfcol=c(5,2), mar=c(2.3,2.3,.5,.5), mgp=c(1.3, .6, 0))
plot(x_val1)
```

The plots are shown on the next page:



data
predicted
observed prob
theoretical prob
Coord Y
Coord X
Density
data − predicted
std residuals
data − predicted
predicted
std residuals
predicted
data − predicted
data
std residuals
data

The `xvalid` function can take the argument `reest=TRUE`. In this case, the variogram will be re-estimated each time a data point is omitted. The argument `variog.obj` should also be included. Here is the command:

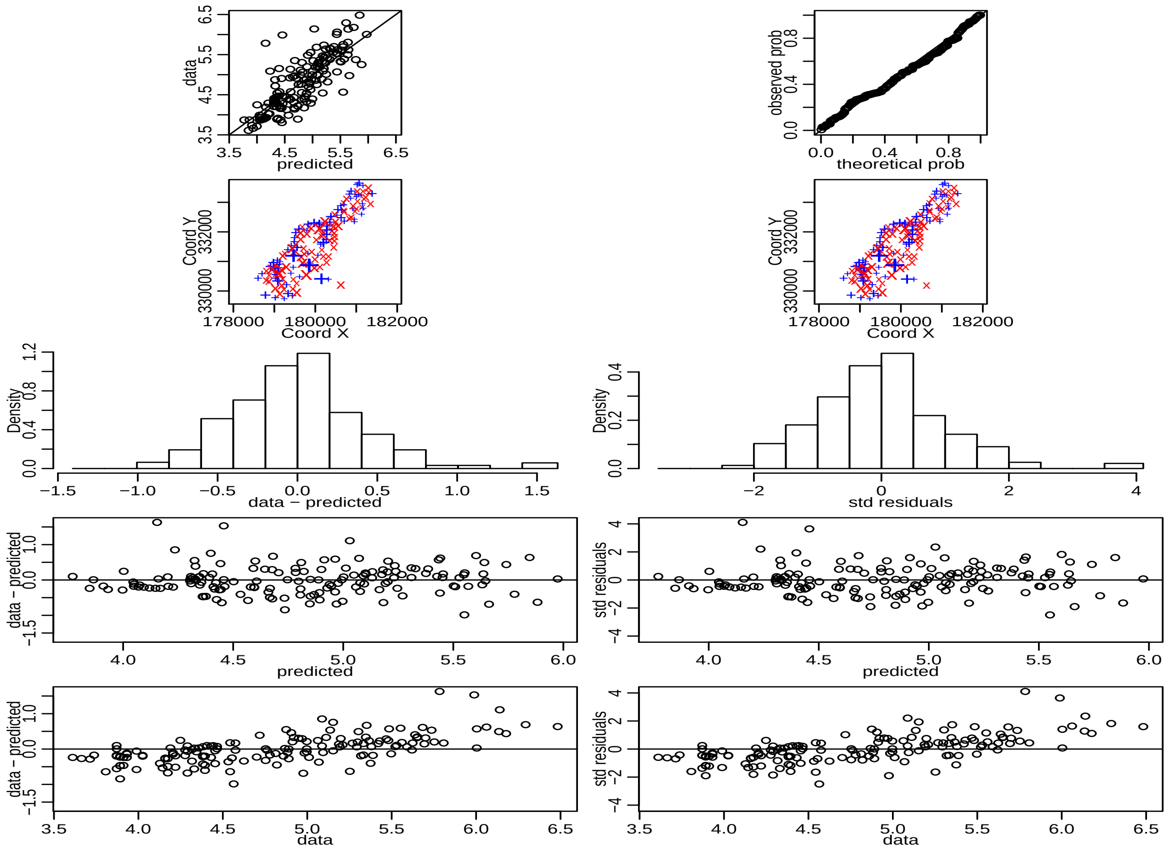
```
x_val2 <- xvalid(bbb, model=fit1, reest=TRUE, variog.obj=var1)
```

Of course this is a time-consuming procedure. The differences between `x_val1$predicted`, `x_val2$predicted` are very small:

```
> compare <- cbind(x_val1$predicted, x_val2$predicted)
> compare[1:5,]
         [,1]     [,2]
[1,] 5.387315 5.385838
[2,] 5.455202 5.453793
[3,] 5.231388 5.230761
[4,] 5.076789 5.076716
[5,] 4.797125 4.797372
```

And we obtain almost the same plot as before with `x_val1`:

```
par(mfcol=c(5,2), mar=c(2.3,2.3,.5,.5), mgp=c(1.3, .6, 0))
plot(x_val2)
```


data
predicted
observed prob
theoretical prob
Coord Y
Coord X
Density
data − predicted
std residuals
predicted
data − predicted
std residuals
data