Applied geostatistics
Exercise 7: Geostatistical risk mapping

D G Rossiter
University of Twente, Faculty of Geo-Information Science & Earth Observation (ITC)

June 26, 2010

Contents

1 Introduction **1**

2 Non-parametric risk mapping **2**
2.1 Indicator variables . . . 2
2.2 Indicator variograms . . . 4
2.3 Indicator kriging . . . 7
2.4 Validation . . . 10
2.5 Cross-validation . . . 15
2.6 Answers . . . 17

3 Parametric risk mapping **19**
3.1 Confidence level maps . . . 21
3.2 Probability-of-exceedence maps . . . 23
3.3 Validation of parametric risk mapping . . . 25
3.4 Answers . . . 28

4 Conclusion **29**
4.1 Answers . . . 29

5 Self-test **30**

References **32**

Version 2.2 Copyright © 2007–2009 ITC All rights reserved. Reproduction and dissemination of the work as a whole (not parts) freely permitted if this original copyright notice is included. Sale or placement on a web site where payment must be made to access this document is strictly prohibited.
To adapt or translate please contact the author (http://www.itc.nl/personal/rossiter).

Index of R concepts 33

盖棺论定 (Gài guān lùn dìng)
"Wait until all the facts are known before judging"
– Chinese proverb

1 Introduction

The essential idea of geostatistical risk mapping is to assign a probability to the occurence of some condition. In this exercise we treat just one aspect: the probability of not exceeding a defined threshold value.

After completing this exercise you should be able to:

1. Convert numeric variables to indicator variables;
2. Compute and model indicator variograms;
3. Predict the probability of exceeding a threshold by indicator kriging;
4. (Optionally) Predict the probability of exceeding a threshold by ordinary kriging and the computation of confidence intervals and probabilities;
5. Validate predictions of exceeding a threshold.

In this exercise we continue with the Jura soil sample dataset introduced in Exercise 2 and manipulated in Exercise 4. A comprehensive risk analysis using this dataset is reported by Goovaerts et al. [2]; the textbook of Goovaerts [1] also covers this in detail.

Task 1 : Load the saved datasets from Exercise 4 §5. This should include:

1. Calibration points `jura.cal`
2. Validation points `jura.val`
3. Prediction grid `jura.grid`

as spatial objects. •

```
> load("JuraEx4.RData")
> ls()

> str(jura.cal)

Formal class 'SpatialPointsDataFrame' [package "sp"] with 5 slots
  ..@ data       :'data.frame':        259 obs. of  9 variables:
  .. ..$ Rock: Factor w/ 5 levels "Argovian","Kimmeridgian",..: 3 2 3 2 5 5 5 1 1 3 ...
  .. ..$ Land: Factor w/ 4 levels "Forest","Pasture",..: 3 2 2 3 3 3 3 3 3 3 ...
  .. ..$ Cd  : num [1:259] 1.74 1.33 1.61 2.15 1.56 ...
  .. ..$ Cu  : num [1:259] 25.72 24.76 8.88 22.7 34.32 ...
  .. ..$ Pb  : num [1:259] 77.4 77.9 30.8 56.4 66.4 ...
  .. ..$ Co  : num [1:259] 9.32 10 10.6 11.92 16.32 ...
  .. ..$ Cr  : num [1:259] 38.3 40.2 47 43.5 38.5 ...
  .. ..$ Ni  : num [1:259] 21.3 29.7 21.4 29.7 26.2 ...
  .. ..$ Zn  : num [1:259] 92.6 73.6 64.8 90 88.4 ...
  ..@ coords.nrs : int [1:2] 1 2
  ..@ coords     : num [1:259, 1:2] 2.39 2.54 2.81 4.31 4.38 ...
```

```
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : NULL
.. .. ..$ : chr [1:2] "X" "Y"
..@ bbox       : num [1:2, 1:2] 0.626 0.58 4.92 5.69
.. ..- attr(*, "dimnames")=List of 2
.. .. ..$ : chr [1:2] "X" "Y"
.. .. ..$ : chr [1:2] "min" "max"
..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slots
.. .. ..@ projargs: chr NA
```

2 Non-parametric risk mapping

The first approach is **non-parametric**. It works with **indicator** variables and directly maps the **probability** of the indicator being true.

2.1 Indicator variables

We continue with cobalt concentration.

An **indicator** for a continuous variable is 1 if the value is below a defined threshold, 0 otherwise.

Task 2 : Compute an indicator variable for threshold 12 mg kg^{-1} Co. Display a postplot of the indicator. •

We make use of a **logical operator**, here `<`. The expression `(jura.cal$Co < 12)` is a vector of `TRUE` and `FALSE` variables, equal in length to the source vector `jura.cal$Co`.

```
> i12 <- (jura.cal$Co < 12)
> summary(i12)

   Mode   FALSE    TRUE    NA's
logical      65     194       0

> sum(i12)/length(i12)

[1] 0.74903
```

Q1 : *How many of the observations are below the threshold? What proportion is this?* *Jump to A1*
•

With the indicator variable in hand, we can examine its spatial distribution. However, it must be part of a spatial structure.

Task 3 : Build a spatial points dataframe with

1. coördinates
2. Co concentration

3. an indicator of whether the Co is below the 12 mg kg^{-1} threshold or not.

•

First, we extract just the Co from the calibration dataset and build a data frame with this and the indicator in two formats: (1) logical and (2) numeric (using the `as.numeric` casting method).

> **Note:** The reason for having the indicator also in numeric form is that some methods do not recognize logical variables.

```
> jura.cal.ind <- data.frame(Co = jura.cal$Co, i12 = i12,
+     i12n = as.numeric(i12))
> str(jura.cal.ind)

'data.frame':        259 obs. of  3 variables:
 $ Co  : num  9.32 10 10.6 11.92 16.32 ...
 $ i12 : logi  TRUE TRUE TRUE TRUE FALSE TRUE ...
 $ i12n: num  1 1 1 1 0 1 0 1 1 0 ...

> rm(i12)
```

> **Note:** We removed vector `i12` from the workspace; it is no longer needed because it has been added to the data frame `jura.cal.ind`; in addition, it is not good practice to have the same name for a workspace object and a field in a dataframe also in the workspace; there can be confusion when trying to find the right object.

Then we add coördinates with the `coordinates` method; these are extracted from the calibration dataset, which parallels the data frame we just built, with the same `coordinates` method:

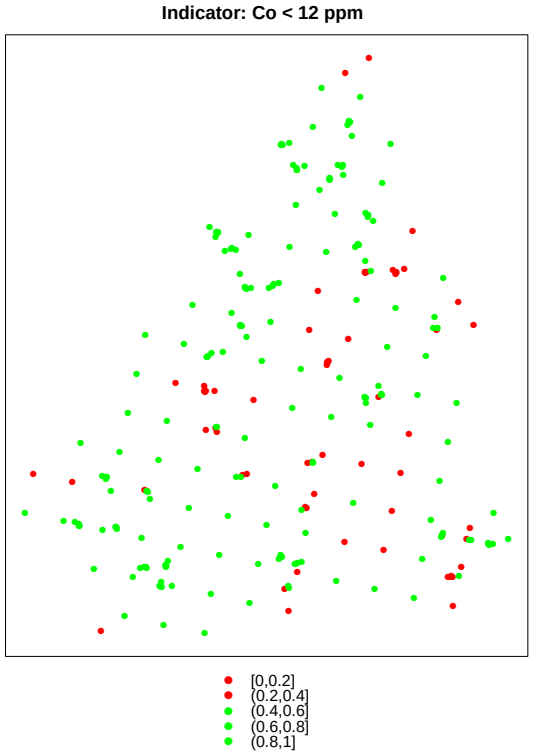
```
> coordinates(jura.cal.ind) <- coordinates(jura.cal)
> str(jura.cal.ind)

Formal class 'SpatialPointsDataFrame' [package "sp"] with 5 slots
  ..@ data       :'data.frame':        259 obs. of  3 variables:
  .. ..$ Co  : num [1:259] 9.32 10 10.6 11.92 16.32 ...
  .. ..$ i12 : logi [1:259] TRUE TRUE TRUE TRUE FALSE TRUE ...
  .. ..$ i12n: num [1:259] 1 1 1 1 0 1 0 1 1 0 ...
  ..@ coords.nrs : num(0)
  ..@ coords     : num [1:259, 1:2] 2.39 2.54 2.81 4.31 4.38 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : NULL
  .. .. ..$ : chr [1:2] "X" "Y"
  ..@ bbox       : num [1:2, 1:2] 0.626 0.58 4.92 5.69
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : chr [1:2] "X" "Y"
  .. .. ..$ : chr [1:2] "min" "max"
  ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slots
  .. .. ..@ projargs: chr NA
```

Q2 : *Why does the data type of object* `jura.cal.ind` *change from* `data.frame` *to* `SpatialPointsDataFrame`? *Jump to A2* •

Task 4 : Plot the locations of the observations, coloured by the indicator value: red for `FALSE` and green for `TRUE`. •

> **Note:** Note that `spplot` can not directly plot an indicator (logical variable); the numeric equivalent must be used.

```
> print(spplot(jura.cal.ind, zcol = "i12n", col.regions = c("red",
+     "green"), main = "Indicator: Co < 12 ppm"))
```



Q3 : *Does this indicator show any spatial structure? If so, what?* *Jump to A3* •

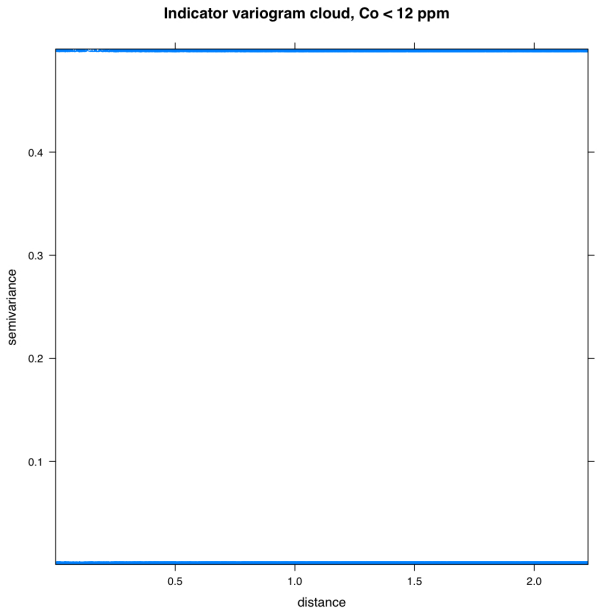
2.2 Indicator variograms

To examine the local spatial dependence, we compute the variogram.

Task 5 : Compute the indicator variogram `cloud` and plot it. •

As usual we use the `variogram` method, but with the `cloud=T` optional argument:

```
> vi <- variogram(i12 ~ 1, loc = jura.cal.ind, cloud = T)
> print(plot(vi, main = "Indicator variogram cloud, Co < 12 ppm"))
```



Q4 : *What are the semivariances of the point-pairs? Explain why.* *Jump to A4* •

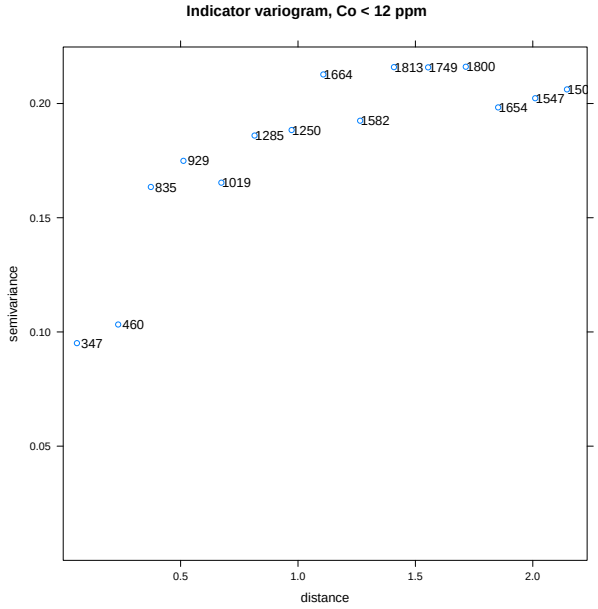
Clearly, an indicator variogram cloud is impossible to interpret; we only show it here to emphasize that only two values of semivariance can be obtained.

These two extremes can be averaged with the empirical variogram.

Task 6 : Compute the **empirical** indicator variogram and plot it. •

We use the `variogram` method:

```
> vi <- variogram(i12 ~ 1, loc = jura.cal.ind)
> print(plot(vi, pl = T, main = "Indicator variogram, Co < 12 ppm"))
```

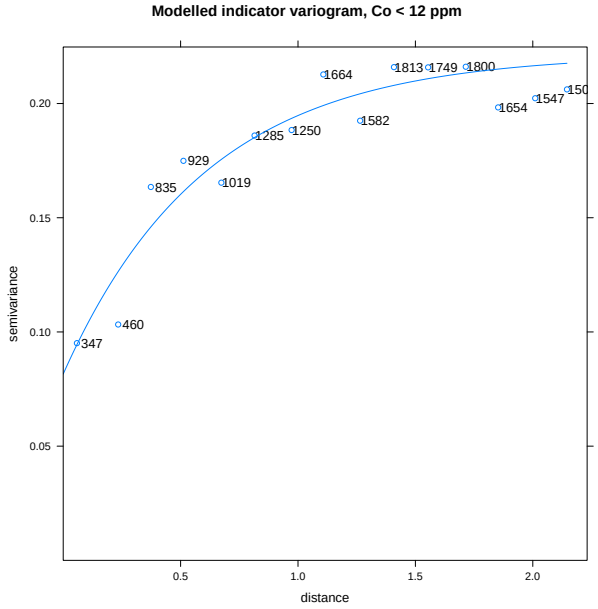


Q5 : *What is the approximate range, sill and nugget of this variogram? What model form might fit it well?* *Jump to A5* •

Task 7 : Model the variogram. •

As before, we construct a model with the `vgm` method and immediately fit it with the `fit.variogram` method:

```
> (vimf <- fit.variogram(vi, vgm(0.15, "Exp", 0.8, 0.07)))

  model    psill   range
1   Nug 0.081583 0.00000
2   Exp 0.140108 0.60656

> print(plot(vi, pl = T, main = "Modelled indicator variogram, Co < 12 ppm",
+     model = vimf))
```



Q6 : *What are the fitted model parameters? What is the effective range? Does the fit appear to be consistent with the empirical variogram?* *Jump to A6* •

```
> vimf[2, "range"] * 3
```

```
[1] 1.8197
```

Q7 : *What is the nugget as a proportion of the total sill? Why is this so high?* *Jump to A7* •

```
> vimf[1, "psill"]/sum(vimf[, "psill"])
```

```
[1] 0.36800
```

2.3 Indicator kriging

With the modelled variogram we can now predict the **probability** of a `TRUE` indicator at any location.

Task 8 : Predict the probability of the indicator over the prediction grid •

```
> ki <- krige(i12 ~ 1, loc = jura.cal.ind, newdata = jura.grid,
+     model = vimf)
```

```
[using ordinary kriging]
```

```
> summary(ki)
```

```
Object of class SpatialPixelsDataFrame
Coordinates:
   min  max
```

```
s1 0.3 5.10
s2 0.4 5.75
Is projected: NA
proj4string : [NA]
Number of points: 10272
Grid attributes:
   cellcentre.offset cellsize cells.dim
s1             0.325     0.05        96
s2             0.425     0.05       107
Data attributes:
   var1.pred        var1.var
 Min.   :0.120   Min.   :0.0982
 1st Qu.:0.623   1st Qu.:0.1327
 Median :0.743   Median :0.1493
 Mean   :0.729   Mean   :0.1647
 3rd Qu.:0.861   3rd Qu.:0.2017
 Max.   :1.011   Max.   :0.2341
```

Q8 : *What is the range of values predicted? What is the theoretical range?* *Jump to A8* •

We see that some values are predicted with $p > 1$, which of course is not meaningful.

Task 9 : Limit the predicted probabilities to the range $[0..1]$. •

For this we use the `pmin` ("parallel minimum") method; and just to be sure about the lower limit, the `pmax` ("parallel maximum") method:

```
> ki$var1.pred <- pmin(1, ki$var1.pred)
> ki$var1.pred <- pmax(0, ki$var1.pred)
> summary(ki@data)

   var1.pred        var1.var
 Min.   :0.120   Min.   :0.0982
 1st Qu.:0.623   1st Qu.:0.1327
 Median :0.743   Median :0.1493
 Mean   :0.729   Mean   :0.1647
 3rd Qu.:0.861   3rd Qu.:0.2017
 Max.   :1.000   Max.   :0.2341
```

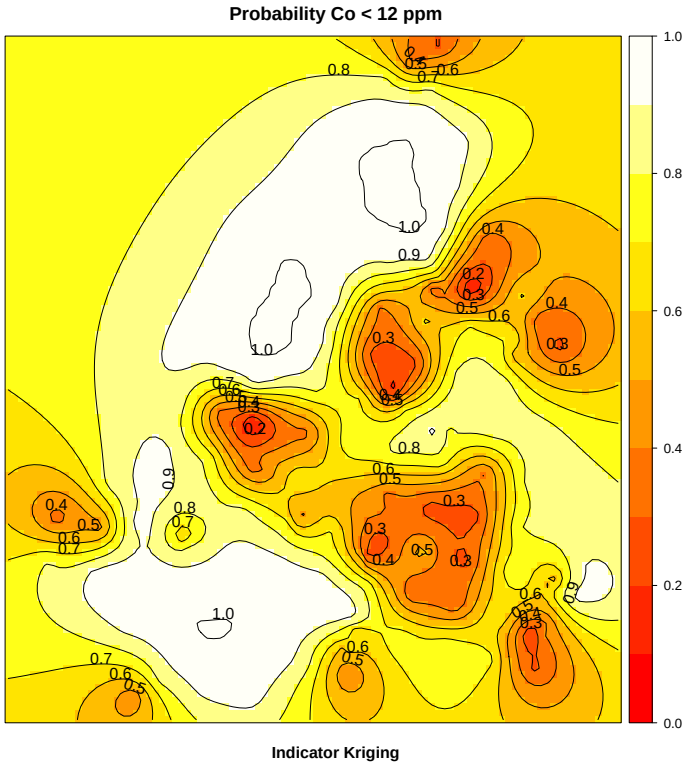
Task 10 : Plot the probability of the indicator. •

We use the `spplot` method with a different colour scheme to emphasize that this is a different kind of kriging output than what we've been viewing previously. We also specify the sequence of colours, to see the full $[0, 1]$ range.

This figure will be used in the following section, so we save it as a lattice graphics object.

```
> plot.ik12 <- spplot(ki, zcol = "var1.pred", at = seq(0,
+     1, by = 0.1), col.regions = heat.colors(64), main = "Probability Co < 12 ppm",
```

```
+     sub = "Indicator Kriging", contour = T)
> print(plot.ik12)
```

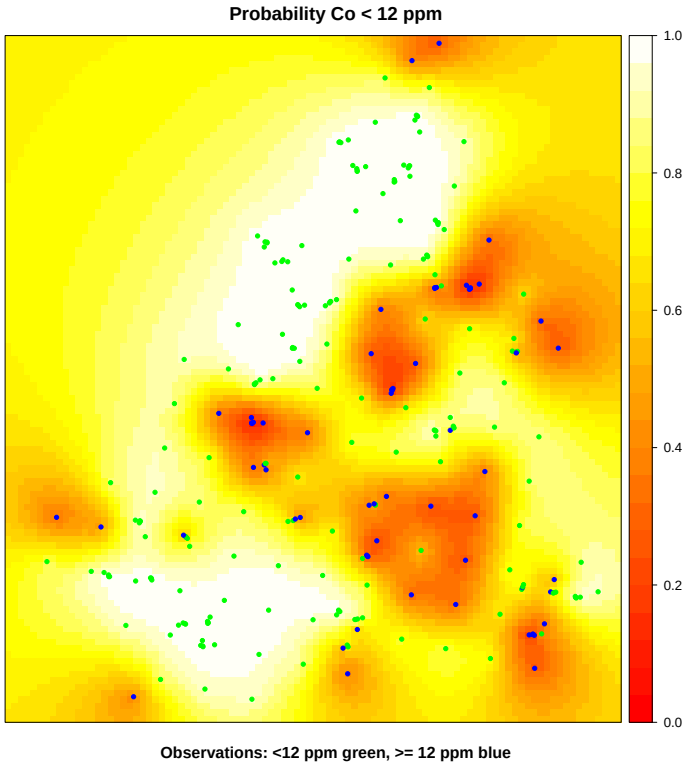


Q9 : *Describe the spatial pattern of the predicted probabilities.* *Jump to A9* •

Q10 : *What is the predicted value away from the sample points, e.g. in the NW corner? Why this value?* *Jump to A10* •

We may get more insight by over-printing the indicators, by making use of the `sp.layout` optional argument to add layout elements; these must first be built as lists. Here we plot the `TRUE` indicators in green, the `FALSE` in blue, using the `ifelse` method to choose which.

```
> layout.ind.pts <- list("sp.points", jura.cal.ind, col = ifelse(jura.cal.ind$i12,
+     "green", "blue"), pch = 20)
> print(spplot(ki, zcol = "var1.pred", at = seq(0, 1, by = 0.04),
+     col.regions = heat.colors(64), main = "Probability Co < 12 ppm",
+     sub = "Observations: <12 ppm green, >= 12 ppm blue",
+     sp.layout = list(layout.ind.pts)))
> rm(layout.ind.pts)
```



Q11 : *Do you have any new insights into the patterns, now that the positions and indicator values of the calibration points are shown?* *Jump to A11* •

Task 11 : Remove temporary objects from the workspace. •

We keep the fitted model `vimf` and the lattice graphics object `plot.ik12` for the next section.

```
> rm(vi, ki)
```

2.4 Validation

We've held out 100 other observations, so we can compare the predicted **probability** that the indicator is `TRUE` with the actual indicator, i.e. the Co value at the validation points classified by the 12 mg kg^{-1} threshold.

Task 12 : Predict the probability that the indicator is `TRUE` at the 100 validation points. •

We repeat the indicator kriging, but now at the validation points instead of the prediction grid:

```
> ki.val <- krige(i12 ~ 1, loc = jura.cal.ind, newdata = jura.val,
+     model = vimf)

[using ordinary kriging]
```

```
> summary(ki.val)

Object of class SpatialPointsDataFrame
Coordinates:
    min   max
X 0.491 4.745
Y 0.524 5.285
Is projected: NA
proj4string : [NA]
Number of points: 100
Data attributes:
   var1.pred        var1.var
 Min.   :0.212   Min.   :0.101
 1st Qu.:0.573   1st Qu.:0.129
 Median :0.754   Median :0.137
 Mean   :0.732   Mean   :0.136
 3rd Qu.:0.920   3rd Qu.:0.141
 Max.   :1.004   Max.   :0.171
```

Task 13 : Compute the indicator at the validation points. •

The indicator is obtained by thresholding the Co values, as we did above for the calibration set:

```
> i12.val <- (jura.val$Co < 12)
> summary(i12.val)

   Mode   FALSE    TRUE    NA's
logical      31      69       0
```

Q12 : *What proportion of the validation points are below the threshold? How does this compare with the calibration points? Why the difference?* *Jump to A12* •

```
> sum(i12.val)/length(i12.val)

[1] 0.69

> sum(jura.cal.ind$i12)/length(jura.cal.ind$i12)

[1] 0.74903
```

Task 14 : Compare the actual indicators with the predicted probabilities. •

To do this, we make a data frame with the two fields, and then sort it by probability. We hope that the `TRUE` indicators are associated with high probabilities of being `TRUE`. Note that both vectors, one from the validation set and one from the kriging prediction, have the same order of points, because the kriging prediction is on the same validation set.

We use the `data.frame` method to make the frame. We include the row names (observation number) from the kriging results and the coordinates for reference, in case we need to identify a specific observation:

```
> compare <- data.frame(id = as.numeric(row.names(ki.val@data)),
+     i12 = i12.val, pred = ki.val$var1.pred)
> coordinates(compare) <- coordinates(ki.val)
> str(compare)

Formal class 'SpatialPointsDataFrame' [package "sp"] with 5 slots
  ..@ data       :'data.frame':        100 obs. of  3 variables:
  .. ..$ id  : num [1:100] 1 2 3 4 5 6 7 8 9 10 ...
  .. ..$ i12 : logi [1:100] TRUE TRUE FALSE TRUE TRUE TRUE ...
  .. ..$ pred: num [1:100] 0.982 0.995 0.82 0.658 0.886 ...
  ..@ coords.nrs : num(0)
  ..@ coords     : num [1:100, 1:2] 2.67 3.59 4.01 2.94 1.41 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : NULL
  .. .. ..$ : chr [1:2] "X" "Y"
  ..@ bbox       : num [1:2, 1:2] 0.491 0.524 4.745 5.285
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : chr [1:2] "X" "Y"
  .. .. ..$ : chr [1:2] "min" "max"
  ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slots
  .. .. ..@ projargs: chr NA
```

To sort the frame we use the `order` method on the target field `pred` to find the sequence of rows that would sort this from lowest to highest. We then use that as a row subscript to re-order all the rows.

```
> order(compare$pred)

  [1]  93  73  43  52  47  89  27  10  97  18   9  34  37  99  49  23
 [17]  64  46  16  58  30  76   7  32  36  66  50  87  62  86  51  60
 [33]  41  94  35  13  90  61   4  40  88  45  28  21  17  54  82  63
 [49]   8  96  29  12  72  65  59  55  84  95   3  98  81  20  53  71
 [65]  83  19  38  15  14   5   6  80  79  78  56  75  69  85  42  11
 [81] 100  39  25  57  31  26  74   1  77  48  68  92   2  33  44  91
 [97]  70  67  24  22

> compare.sorted <- compare[order(compare$pred), ]
> str(compare.sorted)

Formal class 'SpatialPointsDataFrame' [package "sp"] with 5 slots
  ..@ data       :'data.frame':        100 obs. of  3 variables:
  .. ..$ id  : num [1:100] 93 73 43 52 47 89 27 10 97 18 ...
  .. ..$ i12 : logi [1:100] FALSE FALSE FALSE TRUE FALSE TRUE ...
  .. ..$ pred: num [1:100] 0.212 0.311 0.333 0.348 0.348 ...
  ..@ coords.nrs : num(0)
  ..@ coords     : num [1:100, 1:2] 3.29 3.75 2.1 3.68 3.36 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : NULL
  .. .. ..$ : chr [1:2] "X" "Y"
  ..@ bbox       : num [1:2, 1:2] 0.491 0.524 4.745 5.285
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : chr [1:2] "X" "Y"
  .. .. ..$ : chr [1:2] "min" "max"
  ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slots
  .. .. ..@ projargs: chr NA
```

Q13 : *Which of the original 100 validation observations has the least predicted probability of being below the threshold? The greatest? Do these*

have the expected indicator values? *Jump to A13* •

We know the last row is number 100, but it is more elegant to get the length of a column vector from the first matrix dimension stored with the object, using the `dim` method:

```
> compare.sorted[1, ]

      coordinates id   i12    pred
93 (3.287, 3.061) 93 FALSE 0.21158

> compare.sorted[dim(compare.sorted)[1], ]

      coordinates id  i12   pred
22 (3.243, 4.518) 22 TRUE 1.0039
```

Task 15 : Display the predicted probabilities and actual indicator value, sorted by predicted probabilities. •

The `order` method preserves the row numbers of the original (non-sorted) frame; in this case we also want to see the row number as sorted. Since we saved the original row number in the `id` field, we can over-write the row names, with the `row.names` method. Note that for spatial objects the concept of row names applies to the included data frame in the `@data` slot of the object.

```
> head(row.names(compare.sorted@data))

[1] "93" "73" "43" "52" "47" "89"

> row.names(compare.sorted@data) <- 1:dim(compare.sorted)[1]
> head(row.names(compare.sorted@data))

[1] "1" "2" "3" "4" "5" "6"
```

Now we can compare the predicted probability and actual indicator side-by-side:

```
> compare.sorted@data[, c("i12", "pred")]

    i12    pred
1   FALSE 0.19030
2    TRUE 0.29967
3   FALSE 0.43042
4   FALSE 0.44198
5    TRUE 0.47900
6   FALSE 0.48861
...
95   TRUE 0.95909
96   TRUE 0.97643
97   TRUE 0.99437
98   TRUE 0.99927
99   TRUE 0.99995
100  TRUE 0.99995
```

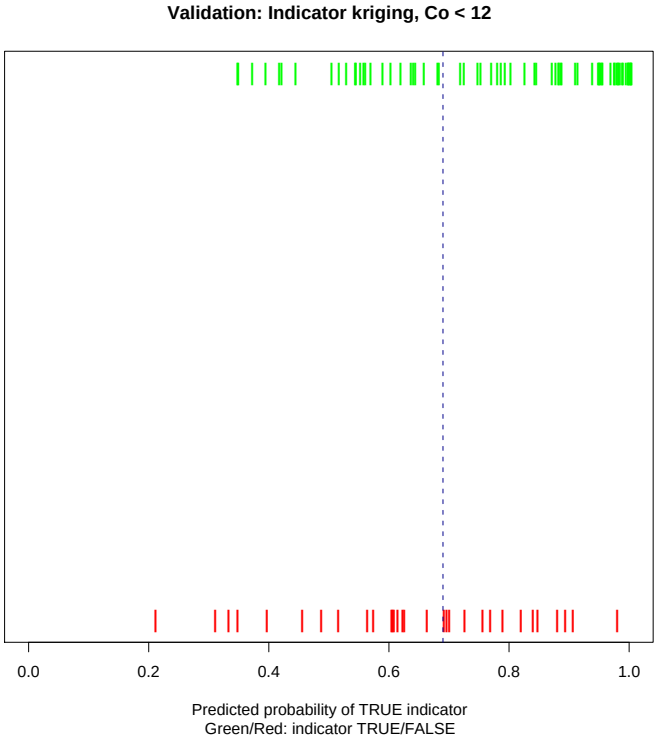
Q14 : *If the indicator kriging were completely successful, what form would this table have? Does it?* *Jump to A14* •

We can see this more easily in a graph:

Task 16 : Plot the T/F values against the predicted probability. •

Each observation is plotted along the x-axis according to its probability; on the y-axis it is either at the top in green if the actual indicator is `TRUE`, otherwise at the bottom in red.

```
> plot(compare$i12 ~ compare$pred, pch = "|", cex = 1.5,
+     xlab = "Predicted probability of TRUE indicator",
+     xlim = c(0, 1), yaxt = "n", ylab = "", col = ifelse(compare$i12,
+         "green", "red"), main = "Validation: Indicator kriging, Co < 12",
+     sub = "Green/Red: indicator TRUE/FALSE")
> abline(v = sum(i12.val)/length(i12.val), lty = 2, col = "darkblue")
```



Q15 : *If the indicator kriging were completely successful, what would this graph look like? Does it? What does this validation say about the indicator kriging's validity in this case?* *Jump to A15* •

Task 17 : Remove temporary objects from the workspace. •

We will use the variogram model in the next section, so don't delete it.

```
> rm(jura.cal.ind, i12.val, compare, compare.sorted)
```

2.5 Cross-validation

This **optional** section uses cross-validation to assess the quality of an Indicator Kriging prediction.

The cross-validation method we used in OK (Exercise 6 §3) can also be applied for indicator kriging. It works the same as for parametric kriging: hold one point out, predict its probability of a `TRUE` indicator from the other points, and then compare this probability with the actual value of the indicator.

Task 18 : Compute the Leave-one-out cross-validation (LOOCV) of the IK prediction of Co concentration less than 12 mg kg^{-1}. •

```
> jura.cal.ind <- data.frame(i12 = (jura.cal$Co < 12))
> coordinates(jura.cal.ind) <- coordinates(jura.cal)
> k.cv <- krige.cv(i12 ~ 1, loc = jura.cal.ind, model = vimf)

> summary(k.cv@data)

   var1.pred        var1.var      observed          residual
 Min.   :0.149   Min.   :0.101   Mode :logical   Min.   :-0.95892
 1st Qu.:0.576   1st Qu.:0.103   FALSE:65        1st Qu.:-0.07993
 Median :0.839   Median :0.109   TRUE :194       Median : 0.03514
 Mean   :0.747   Mean   :0.122   NA's :0         Mean   : 0.00197
 3rd Qu.:0.965   3rd Qu.:0.145                   3rd Qu.: 0.17354
 Max.   :1.011   Max.   :0.177                   Max.   : 0.84175
     zscore              fold
 Min.   :-2.96367   Min.   :  1.0
 1st Qu.:-0.24737   1st Qu.: 65.5
 Median : 0.10297   Median :130.0
 Mean   : 0.00288   Mean   :130.0
 3rd Qu.: 0.51159   3rd Qu.:194.5
 Max.   : 2.46171   Max.   :259.0
```

Q16 : *Which field has the predicted probability of a `TRUE` indicator? Which has the actual indicator?* *Jump to A16* •

Again we see impossible predicted probabilities.

Task 19 : Limit the predicted probabilities to the range $[0 \ldots 1]$. •

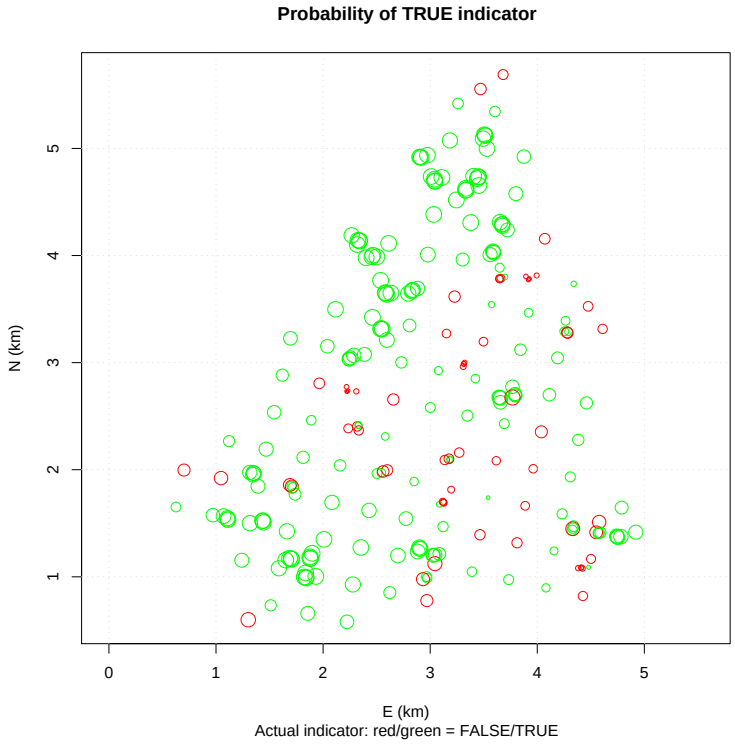
```
> k.cv$var1.pred <- pmin(1, k.cv$var1.pred)
```

We can make an interesting visualization of the cross-validation.

Task 20 : Make a post-plot of the predicted probabilities, with the symbol size proportional to the probability, with the points coloured red for a `FALSE` indicator and green for a `TRUE` indicator. •

We add a small number to the symbol size so that probabilities close to zero can be more easily seen. There is no need to normalize the symbol size because the maximum is by definition 1.

```
> plot(coordinates(k.cv), asp = 1, col = ifelse(k.cv$observed,
+     "green", "red"), cex = 0.2 + 2 * k.cv$var1.pred,
+     xlab = "E (km)", ylab = "N (km)", main = "Probability of TRUE indicator",
+     sub = "Actual indicator: red/green = FALSE/TRUE")
> grid()
```



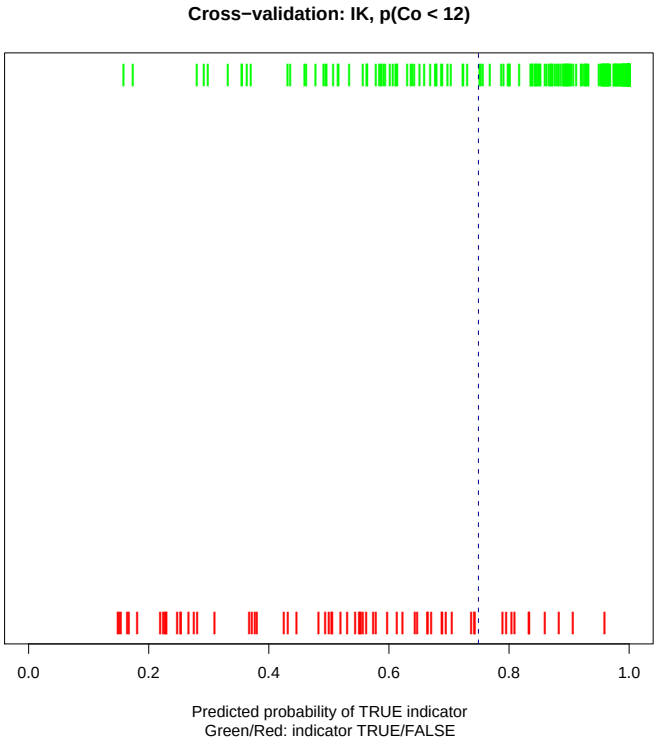
Q17 : *If the model were successful, what would this plot look like? Does it?* *Jump to A17* •

We can visualize the success with the T/F vs. probability plot, as in the independent validation.

Task 21 : Compare the actual indicators with the predicted probabilities, by plotting the T/F values against the predicted probability. •

We have both of these already in the `k.cv` object.

```
> plot(k.cv$observed ~ k.cv$var1.pred, pch = "|", cex = 1.5,
+     xlab = "Predicted probability of TRUE indicator",
+     xlim = c(0, 1), yaxt = "n", ylab = "", col = ifelse(k.cv$observed,
+         "green", "red"), main = "Cross-validation: IK, p(Co < 12)",
+     sub = "Green/Red: indicator TRUE/FALSE")
> abline(v = sum(k.cv$observed)/length(k.cv$observed),
+     lty = 2, col = "darkblue")
```



Q18 : *Assess this model from its cross-validation, as we did for its independent validation. Do the two validation methods agree?* *Jump to A18*
•

Task 22 : Remove temporary objects from the workspace. •

```
> rm(k.cv, jura.cal.ind)
```

2.6 Answers

A1 : *194 (of 259); this is 74.9%.* *Return to Q1* •

A2 : *A data frame with coördinates assigned by the* `coordinates` *method is automatically promoted to a spatial object. A bounding box is also computed.* *Return to Q2* •

A3 : *Yes, the relatively few* `FALSE` *values are somewhat clustered; they are not spread equally over the map. However, there are some locations with* `TRUE` *and* `FALSE` *very close to each other.* *Return to Q3* •

A4 : *Either 0 or 0.5. If the two points have the* **same** *indicator (both 0 or both 1), their difference is 0, so their semivariance is 0. If the two points have different* **same**

indicators (0 and 1, or 1 and 0) their difference is 1, so is their squared difference, and half of this is 0.5. *Return to Q4* •

A5 : *The total sill appears to be about 0.22, the range about 1.1 km, and the nugget 0.07, although the behaviour at very short separation is difficult to model, especially the sharp jump between the second and third bin. An exponential model can "split the difference" between these. The structural sill of the exponential model is then 0.22 - 0.07 = 0.15.* *Return to Q5* •

A6 : *The exponential model has a structural sill of 0.14; combined with the nugget of 0.082; this is a total sill of 0.222. The range parameter is 0.6066 km, which means that the effective range is about 1.82 km (recall: the effective range of the the exponential model is 3 times the range parameter; here the variogram reaches 95% of the sill value.*

The fit "splits the difference" fairly well, especially between the second and third bins. *Return to Q6* •

A7 : *36.8% of the total variance is represented by the nugget; this is variability that is* ***unexplained*** *even at the shortest range. This reflects the several places noted above, where* `TRUE` *and* `FALSE` *values are very close to each other in geographic space.* *Return to Q7* •

A8 : *The actual range is* $[0.120 \ldots 1.011]$*; the theoretical range is* $[0 \ldots 1]$*, i.e. a probability; in this case the actual range does not reach* $p = 0$ *anywhere, but exceeds* $p = 1$ *in some locations. This probability is of course meaningless, but is a possible result of a non-convex predictor like OK. We can just interpret these as* $p = 1$*, or, if we wish, limit them (see next task).* *Return to Q8* •

A9 : *Clear "hot spots" of low indicator probability (i.e. high Co concentration) and smooth transitions to larger patches of high indicator probability. Almost the same probability (no patches) in the NW corner.* *Return to Q9* •

A10 : *Away from the data points the prediction is uniform, 0.749. This is the proportion of* `TRUE` *indicators in the calibration data set.* *Return to Q10* •

A11 : *There is a smooth transition of probabilities between nearby single or clusters of* `TRUE` *and* `FALSE` *indicators.* *Return to Q11* •

A12 : *Only 69% are below the threshold; in the calibration set that was almost 75%. So the validation set has a smaller proportion of* `TRUE` *indicators at this threshold, i.e. a higher proportion of high-Co points. There does not seem to be a particular reason; it's by chance for small samples.* *Return to Q12* •

A13 : *Least: observation 93 (*$p = 0.2116$*); greatest: 22 (*$p = 1.004$*); these have*

the expected `FALSE` *and* `TRUE` *indicators, respectively.* *Return to Q13* •

A14 : *The 39* `FALSE` *values would be the first 39 entries of the table, next to their prediction probabilities; these would all be less than the prediction probabilities for the 61* `TRUE` *values. This is certainly not the case. The second sorted record is already (incorrectly)* `TRUE`*; and* `FALSE` *values occur up to sorted record 87 (*$p = 0.98$*, so the kriging is "almost certain" that this point is* `TRUE` *when in fact it is* `FALSE`*).* *Return to Q14* •

A15 : *All the green points at the top would be on the right side (high predicted probabilites that these* `TRUE` *points are true, and all the red points on the left. There would be a clear probability (vertical line) to separate them, preferably at the overall probability of the indicator in the validation set, here* 0.69*.*

This is clearly not seen. There is a lot of overlap; in particular some `FALSE` *indicators are predicted to be* `TRUE` *with probabilities up to* 0.9*. The region* $p > 0.9$ *is well-modelled (only one false negative), but elsewhere not so well.* *Return to Q15* •

A16 : *Predicted probability in field* `var1.pred`*; actual indicator in* `observed`*.* *Return to Q16* •

A17 : *The red circles would be small and the green ones large. This appears to be the overall pattern but there are some clear exceptions.* *Return to Q17* •

A18 : *This graph looks quite similar to the 100-point independent validation, but with more (259) points. The region with* $p > .9$ *(i.e. almost surely below the threshold) is, again, well-modelled, but elsewhere there is considerable overlap.* *Return to Q18* •

3 Parametric risk mapping

In this **optional** section we look at another way to map the probability of exceeding (or not) a defined threshold. This is simply an extension of the **parametric** kriging approach, i.e. Ordinary Kriging or any of its extensions such as Universal Kriging, and relies on all the assumptions of that approach.

In Exercise 4 §4.3 we discussed the result from probability theory that the two-sided interval which has probability $(1 - \alpha)$ of containing the true value z is:

$$(\hat{z} - t_{\alpha/2,n-1} \cdot s_{\hat{z}}) \;\leq\; z \;\leq (\hat{z} + t_{\alpha/2,n-1} \cdot s_{\hat{z}})$$

where $\hat{z}$ is the estimated value, $t_{\alpha/2,n-1}$ is Student's t with $n-1$ degrees of freedom at confidence level $\alpha/2$ and $s_{\hat{z}}$ is the sample standard deviation; n is the sample size.

Note: The total probability of Type I error α, say 0.05, is halved, say to 0.025 for each side of the interval, because this is a two-sided interval. The t-distribution must be used because we are estimating both the mean and variance from the same sample; for reasonably-large sample sizes the normal distribution itself can be used.

However, there is an important **assumption** behind this formula: the theory of stationary random fields, which states that the actual value at a point is the result of a second-order stationary random process with normally-distributed error at the point. This assumption can never be tested, because we only have one realization of the process. However it should also lead to a normally-distributed **sample**, if the sample is unbiased (e.g. the result of some random sampling design). In practice we visualize this with a histogram of the target variable.

If we can not reasonably make this assumption, the Indicator approach (§2) may still be valid, since it makes no assumptions about the distribution of the realizations at a point.

In the case of OK, the kriging prediction gives us both the **predicted value** $\hat{z}$ and its **prediction variance** $\sigma^2_{\hat{z}}$, which is the square of the standard error of prediction. So we have all the information to compute the confidence interval at **all** prediction points.

We can use this two ways:

1. Fix the **probability of Type I error** α and compute the upper and lower **confidence limits**, i.e. the highest (or lowest) probable values to be found at that location;
2. Fix the **threshold** (as in the indicator approach) and compute the **probability of exceeding** (or not) that threshold; this is directly comprable to the probability map produced by IK at that threshold.

Recall that use the parametric approach (e.g. OK) the target variable should be reasonably well-distributed, in order to estimate the variogram reliably.

Task 23 : Display a histogram of the Co concentration in the Jura calibration dataset. •

```
> hist(jura.cal$Co, col = "lightblue", border = "darkblue",
+     main = "Co concentration, Jura soils", sub = "mg kg-1")
```

Q19 : *Is this distribution symmetric or skewed? Are there any extreme values?* *Jump to A19* •

Task 24 : Model the variogram for Co concentration and predict its value and prediction variance over the grid by Ordinary Kriging. •

This repeats Exercise 4 §4.1 and §4.4.

```
> v <- variogram(Co ~ 1, loc = jura.cal, cutoff = 1.6)
> vmf <- fit.variogram(v, vgm(12.5, "Pen", 1.2, 1.5))
> k.grid <- krige(Co ~ 1, loc = jura.cal, newdata = jura.grid,
+     model = vmf)

[using ordinary kriging]

> summary(k.grid)

Object of class SpatialPixelsDataFrame
Coordinates:
   min  max
s1 0.3 5.10
s2 0.4 5.75
Is projected: NA
proj4string : [NA]
Number of points: 10272
Grid attributes:
   cellcentre.offset cellsize cells.dim
s1             0.325     0.05        96
s2             0.425     0.05       107
Data attributes:
   var1.pred        var1.var
 Min.   : 2.91   Min.   : 1.85
 1st Qu.: 8.48   1st Qu.: 4.26
 Median : 9.73   Median : 5.63
 Mean   : 9.54   Mean   : 7.86
 3rd Qu.:10.98   3rd Qu.:12.24
 Max.   :15.81   Max.   :15.01
```

Now we use the OK prediction in two different ways of assessing the uncertainty in a prediction.

3.1 Confidence level maps

The first method is to fix the probability of Type I error α and compute the upper and lower confidence limits, i.e. the highest (or lowest) probable values to be found at that location. We can then display these upper and lower bounds as maps which can be interpreted by the decision maker as conservation and liberal pictures of what might be found in reality.

Task 25 : Compute the upper and lower confidence limits at all points in the prediction grid, with $\alpha = 0.05$, i.e. $\alpha = 0.025$ in each tail. •

At each point we have a predicted value and its variance. All we need in addition is the appropriate t-value. This depends both on α and the degrees of freedom, which is one less than the sample size, and is obtained with the `qt` ("quantiles of the t-distribution") method:

```
> (t.val <- qt(0.975, length(jura.cal$Co) - 1))

[1] 1.9692
```

Note this is almost the z-value (which we could get with the `qz` ("quantiles of the Normal distribution") method, because the sample is large.

We can then add fields to the kriging object, with the confidence limits:

```
> k.grid$lcl <- k.grid$var1.pred - (t.val * sqrt(k.grid$var1.var))
> k.grid$ucl <- k.grid$var1.pred + (t.val * sqrt(k.grid$var1.var))
> summary(k.grid@data)
```

```
   var1.pred        var1.var          lcl              ucl       
 Min.   : 2.91   Min.   : 1.85   Min.   :-1.23   Min.   : 6.27  
 1st Qu.: 8.48   1st Qu.: 4.26   1st Qu.: 2.08   1st Qu.:13.40  
 Median : 9.73   Median : 5.63   Median : 3.92   Median :15.44  
 Mean   : 9.54   Mean   : 7.86   Mean   : 4.23   Mean   :14.86  
 3rd Qu.:10.98   3rd Qu.:12.24   3rd Qu.: 6.29   3rd Qu.:17.25  
 Max.   :15.81   Max.   :15.01   Max.   :13.08   Max.   :18.83  
```

Q20 : *Are all the upper and lower confidence limits realistic?* *Jump to A20* •

We correct the non-physical values using the `pmax` ("parallel maximum") method (note that the `max` ("maximum") method uses the maximum of the entire vector, not element-wise):

```
> k.grid$lcl <- pmax(k.grid$lcl, 0)
> summary(k.grid@data)
```

```
   var1.pred        var1.var          lcl              ucl       
 Min.   : 2.91   Min.   : 1.85   Min.   : 0.00   Min.   : 6.27  
 1st Qu.: 8.48   1st Qu.: 4.26   1st Qu.: 2.08   1st Qu.:13.40  
 Median : 9.73   Median : 5.63   Median : 3.92   Median :15.44  
 Mean   : 9.54   Mean   : 7.86   Mean   : 4.24   Mean   :14.86  
 3rd Qu.:10.98   3rd Qu.:12.24   3rd Qu.: 6.29   3rd Qu.:17.25  
 Max.   :15.81   Max.   :15.01   Max.   :13.08   Max.   :18.83  
```

Task 26 : Plot the lower limit, predicted value, and upper limit side-by-side with the same colour scale. •

First we determine a common limits and cuts using the `min` and `max` methods; note that we know the minimum must be in field `lcl` and the maximum in field `ucl`; we round the maximum up to the nearest integer with the `ceiling` method and the minimum down to the nearest integer with the `floor` method:

```
> (at.pred <- seq(floor(min(k.grid$lcl)), ceiling(max(k.grid$ucl)),
+     by = 1))
```

```
 [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
```

> **Note:** We already knew that the minimum was zero from the previous task, but we show the calculation anyway for the more general case.

Now we build the three figures and plot them together:

```
> plot.1 <- spplot(k.grid, zcol = "lcl", at = at.pred,
+     contour = T, col.regions = bpy.colors(64), main = "Lower 95% confidence limit",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T))
> plot.2 <- spplot(k.grid, zcol = "var1.pred", at = at.pred,
+     contour = T, col.regions = bpy.colors(64), main = "Predicted values, Co (ppm)",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T))
```

```
> plot.3 <- spplot(k.grid, zcol = "ucl", at = at.pred,
+     contour = T, col.regions = bpy.colors(64), main = "Upper 95% confidence limit",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T))
> print(plot.1, split = c(1, 1, 2, 2), more = T)
> print(plot.2, split = c(1, 2, 2, 2), more = T)
> print(plot.3, split = c(2, 1, 2, 2), more = F)
```

Q21 : *What is the interpretation of the lower confidence level, predicted, and upper confidence level maps?* *Jump to A21* •

The upper and lower confidence interval maps can be **categorized** at any **threshold** value (lower and upper bound, respectively) into a **binary** map that shows areas that **most probably** exceed or not the threshold.

Task 27 : Display a binary map of the areas that might exceed the threshold of 12 mg kg^{-1} Co (same as used in the IK example above). •

These are areas where we are sure, with only a 2.5% chance of being wrong, that the value is not below this threshold. We can get this by reclassifying the upper confidence level map at this value.

```
> k.grid$ucl12 <- (k.grid$ucl >= 12)
> summary(k.grid$ucl12)

   Mode   FALSE    TRUE    NA's
logical    1654    8618       0
```

Q22 : *What proportion of the total grid is almost surely below this threshold?* *Jump to A22*

•

```
> 1 - (sum(k.grid$ucl12)/length(k.grid$ucl12))

[1] 0.16102
```

Now we display the map of the areas possibly above the threshold:

```
> plot(coordinates(k.grid), pch = ifelse(k.grid$ucl12,
+     20, " "), cex = 0.5, asp = 1, col = "red", xlab = "E",
+     ylab = "N", main = "Areas possibly > 12 ppm Co",
+     sub = "95% two-sided confidence interval")
> grid()
```

3.2 Probability-of-exceedence maps

The second method is to fix the threshold and compute the probability of exceeding (or not) it. This uses the t-distribution in the inverse way as the previous subsection. At each prediction point we have the prediction and its variance. From this we can compute how many standard deviations from the predicted value is the target threshold, which we call Z_t.

This is the difference between the threshold and the prediction, normalized by t the kriging standard error of the prediction:

$$w = \frac{Z_t - \hat{z}_0}{\sigma_{\hat{z}_0}}$$

Then the `pt` ("probability of a lower t") method gives the associated probability that this value could be obtained by chance.

For example, for a threshold of 12 mg kg^{-1} Co, at the first prediction point:

```
> target <- 12
> k.grid$var1.var[1]

[1] 15.005

> target - k.grid$var1.pred[1]

[1] 2.2908

> sqrt(k.grid$var1.var[1])

[1] 3.8736

> (w <- (target - k.grid$var1.pred[1])/sqrt(k.grid$var1.var[1]))

[1] 0.59139
```

Q23 : *How far is the threshold above the predicted value at this point? What is the standard deviation? How many standard deviations is the threshold above the predicted value?* *Jump to A23* •

Knowing the degrees of freedom, this can be converted into a one-tail probability (since we are only interested in exceedence), using the `pt` method:

```
> pt(w, length(jura.cal$Co))

[1] 0.72261
```

Q24 : *What is the probability that this point does* not *exceed the threshold?* *Jump to A24* •

Task 28 : For the entire prediction grid, determine and display the probability that the Co concentration is not above 12 mg kg^{-1}. •

```
> k.grid$p12 <- pt((target - k.grid$var1.pred)/sqrt(k.grid$var1.var),
+     length(jura.cal$Co))
> plot.p12 <- spplot(k.grid, zcol = "p12", col.regions = heat.colors(64),
+     at = seq(0, 1, by = 0.1), contour = T, main = "Probability < 12 ppm Co",
+     sub = "Parametric")
> print(plot.p12)
```

Q25 : *How does this map compare with the indicator kriging map at the same threshold from §2.3? Describe any differences. Should they show the same probabilities of exceedence?* *Jump to A25* •

We can answer this more easily by comparing the maps side-by-side, with the lattice object saved from the previous section:

```
> print(plot.p12, split = c(1, 1, 2, 1), more = T)
> print(plot.ik12, split = c(2, 1, 2, 1), more = F)
```

3.3 Validation of parametric risk mapping

We repeat the validation of §2.4 for both methods: confidence intervals and probability of exceedence.

Task 29 : Predict the probability that the indicator is `TRUE` at the 100 validation points. •

We repeat the OK, but now at the validation points instead of the prediction grid:

```
> k.val <- krige(Co ~ 1, loc = jura.cal, newdata = jura.val,
+     model = vmf)

[using ordinary kriging]

> summary(k.val)

Object of class SpatialPointsDataFrame
Coordinates:
    min   max
X 0.491 4.745
Y 0.524 5.285
Is projected: NA
proj4string : [NA]
Number of points: 100
Data attributes:
   var1.pred        var1.var
 Min.   : 3.58   Min.   :1.98
 1st Qu.: 7.70   1st Qu.:4.35
 Median :10.08   Median :4.84
 Mean   : 9.45   Mean   :4.71
 3rd Qu.:11.27   3rd Qu.:4.93
 Max.   :14.01   Max.   :7.92
```

Task 30 : Compute the upper confidence level at $\alpha = 0.05$ at these validation points, and threshold this at the 12 mg kg^{-1} Co level to predict how many points are definitely (at this confidence level) above the threshold. •

```
> (t.val <- qt(0.975, length(jura.cal$Co) - 1))

[1] 1.9692
```

```
> k.val$ucl <- k.val$var1.pred + (t.val * sqrt(k.val$var1.var))
> k.val$ucl12 <- (k.val$ucl >= 12)
> str(k.val@data)

'data.frame':        100 obs. of  4 variables:
 $ var1.pred: num  5.05 8.95 11.18 11.09 8.49 ...
 $ var1.var : num  3.53 4.26 6.56 4.89 6.51 ...
 $ ucl      : num  8.75 13.01 16.22 15.45 13.51 ...
 $ ucl12    : logi  FALSE TRUE TRUE TRUE TRUE FALSE ...
```

Q26 : *What proportion of the validation points are predicted to be definitely above the threshold? What proportion are in fact above it?* *Jump to A26*
•

```
> sum(k.val$ucl12)/length(k.val$ucl12)

[1] 0.73

> sum((jura.val$Co >= 12))/length(jura.val$Co)

[1] 0.31
```

Note that the first proportion depends in part on the confidence level.

Q27 : *If the confidence level is lowered, for example to* $\alpha = 0.5$ *(25% in each tail), does the number of points predicted to be above the threshold increase or decrease? Why?* *Jump to A27* •

```
> (t.val <- qt(0.75, length(jura.cal$Co) - 1))

[1] 0.67544

> k.val$ucl75 <- k.val$var1.pred + (t.val * sqrt(k.val$var1.var))
> k.val$ucl75.12 <- (k.val$ucl75 >= 12)
> sum(k.val$ucl75.12)/length(k.val$ucl75.12)

[1] 0.43
```

Q28 : *At what confidence level, then, should the parametric method be validated? What is the result?* *Jump to A28* •

```
> sum(k.val$var1.pred >= 12)/length(k.val$var1.pred)

[1] 0.11

> sum((jura.val$Co >= 12))/length(jura.val$Co)

[1] 0.31
```

We now validate the parameteric probability method.

Task 31 : Compute the probabilities of non-exceedence for the 100 validation points. •

Note that we follow the convention, also used in Indicator Kriging, that a `TRUE` value is **below** the threshold.

We follow the procedure of §3.2.

```
> k.val$p12 <- pt((12 - k.val$var1.pred)/sqrt(k.val$var1.var),
+     length(jura.cal$Co))
> summary(k.val@data)
```

```
   var1.pred        var1.var          ucl          ucl12
 Min.   : 3.58   Min.   :1.98   Min.   : 6.34   Mode :logical
 1st Qu.: 7.70   1st Qu.:4.35   1st Qu.:11.93   FALSE:27
 Median :10.08   Median :4.84   Median :14.57   TRUE :73
 Mean   : 9.45   Mean   :4.71   Mean   :13.70   NA's :0
 3rd Qu.:11.27   3rd Qu.:4.93   3rd Qu.:15.54
 Max.   :14.01   Max.   :7.92   Max.   :17.86
     ucl75        ucl75.12           p12
 Min.   : 4.53   Mode :logical   Min.   :0.135
 1st Qu.: 9.19   FALSE:57        1st Qu.:0.628
 Median :11.70   TRUE :43        Median :0.788
 Mean   :10.91   NA's :0         Mean   :0.770
 3rd Qu.:12.75                   3rd Qu.:0.977
 Max.   :15.32                   Max.   :1.000
```

Task 32 : Compare the actual indicators with the predicted probabilities. •

We build a data frame with these two:

```
> compare <- data.frame(i12 = (jura.val$Co < 12), p12 = k.val$p12)
> summary(compare)
```

```
   i12              p12
 Mode :logical   Min.   :0.135
 FALSE:31        1st Qu.:0.628
 TRUE :69        Median :0.788
 NA's :0         Mean   :0.770
                 3rd Qu.:0.977
                 Max.   :1.000
```

Task 33 : Plot the T/F values against the predicted probability. •

```
> plot(compare$i12 ~ compare$p12, pch = "|", cex = 1.5,
+     xlab = "Predicted probability of TRUE indicator",
+     xlim = c(0, 1), yaxt = "n", ylab = "", col = ifelse(compare$i12,
+         "green", "red"), main = "Validation: OK, p(Co < 12)",
+     sub = "Green/Red: indicator TRUE/FALSE")
> abline(v = sum(compare$i12)/length(compare$i12), lty = 2,
+     col = "darkblue")
```

Q29 : *If the indicator kriging were completely successful, what would this graph look like? Does it? What does this validation say about the indicator kriging's validity in this case?* *Jump to A29* •

Task 34 : Clean up from this section. •

```
> rm(v, vmf, k.grid, target, t.val, w, plot.p12, plot.ik12,
+     k.val, compare)
```

3.4 Answers

A19 : *Symmetric, no extreme values.* *Return to Q19* •

A20 : *No, the lower confidence limit has some values* < 0. *Return to Q20* •

A21 :

1. *The* ***lower*** *confidence level map shows the* ***lowest*** *value that might be at each location, with only a 2.5% chance that the true value is lower;*
2. *The predicted map shows the* ***most likely*** *value that is expected to be found at each location;*
3. *The* ***upper*** *confidence level map shows the* ***highest*** *value that might be at each location, with only a 2.5% chance that the true value is higher.*

Return to Q21 •

A22 : *Only 16%. This seems quite low; but recall (1) it is a very conservative estimate; (2) because of the high kriging prediction variance in the extrapolation area (outside the area with the sample) the corners of the map have very wide confidence intervals, so must be considered "at risk" in this conservative approach.* *Return to Q22* •

A23 : *The target is higher than the predicted value by 2.29 mg kg*$^{-1}$ *Co. The standard deviation at this point is 3.87 mg kg*$^{-1}$ *Co. So the difference is equivalent to 0.59 standard deviations.* *Return to Q23* •

A24 : $p = 0.72$*; thus there is a 28% chance that it is exceeded.* *Return to Q24* •

A25 : *They should be the same map, arrived at by completely different methods. That is, the probability of exceeding a defined threshold should be the same. But the variograms are different; furthermore prediction points in the OK map are weighted averages of Co values, whereas in the IK map they are weighted averages of 0's and 1's. So it is not surprising there are some differences.*

There are clear differences: the OK/confidence interval/probability map shows tighter "hot spots" and narrower transition zones. It has no areas of $p = 1$*, which were obtained in the IK map by limiting predicted* $p > 1$*, and it has the probabilities in the "hot spots" closer to* $p = 0$*, i.e. more certainty that the threshold*

is exceeded. The parametric method uses more information and this seems to be reflected in the results. *Return to Q25* •

A26 : *The prediction is, with only 2.5% chance that we have missed a higher value, that 73 validation points could be above the threshold; the actual number is 31. This shows clearly that the confidence interval gives a conservative estimate, in the sense that if there is almost any chance of being above the threshold, it so predicts.* *Return to Q26* •

A27 : *It decreases, in fact to 43; but the probability that we have missed a given point that is in fact above the threshold is now 25%, rather than 2.5%. This is a less conservative estimate.* *Return to Q27* •

A28 : *At the expected value (i.e. no confidence interval); this is the expected value. In fact the method predicts only 11 above the threshold, when in fact 31 are.* *Return to Q28* •

A29 : *This is the same graph, with the same meaning, as in the validation of IK. To repeat, all the green points at the top would be on the right side (high predicted probabilites that these* `TRUE` *points are true, and all the red points on the left. There would be a clear probability (vertical line) to separate them, preferably at the overall probability of the indicator in the validation set, here* 0.69.

This is clearly not seen. There is some separation; red points tend to be to the left (lower probability) of the green, but again there is a lot of overlap. The vertical line at the actual probability in the validation set doesn't separate the two groups. *Return to Q29* •

4 Conclusion

Q30 : *What is your overall conclusion about the success of risk mapping in this case (this metal, this threshold, this area)? Give some possible explanations for this.* *Jump to A30*
•

4.1 Answers

A30 : *At this theshold (12 mg kg^{-1}) of this metal (Co), even with a fairly dense set of observations (259), it is not possible to clearly map areas that are probably above vs. probably not above the threshold. This is clearly seen in the high nugget variance in the indicator variogram, as well as the validation results.*

There are two interpretations of this, either of which may be true:

1. *The cutoff is in the middle of the distribution and does not represent a threshold in nature;*

2. *The process is very local, and perhaps non-spatial; that is, there is little or no spatial correlation in the process that produces the high Co values.*

Return to Q30 •

5 Self-test

This section is a small self-test of how well you mastered this exercise. You should be able to complete the tasks and answer the questions with the knowledge you have gained from the exercise. Please **submit your answers (including graphical output) to the instructor** for grading and sample answers.

This self-test repeats the key steps in indicator kriging, but using a different threshold. We saw in the exercise that the 12 mg kg^{-1} Co threshold was somewhat difficult to model, because there were some locations with almost adjacent `TRUE` and `FALSE` indicators. The histogram for cobalt was fairly symmetric and with no obvious long tail of high values. This is not the case for most of the other metals.

If necessary, load the required packages `sp`, `gstat` and `lattice`.

Task 1 : Make a histogram of the lead (Pb) concentrations in the calibration data set. •

Q1 : *Describe the form of this histogram.* •

Task 2 : Suppose the 100 mg kg^{-1} Pb threshold is a regulatory limit (i.e. soils with more Pb are considered polluted); make an indicator variable for this threshold. •

Q2 : *How many of the calibration sample points are below the threshold? What proportion is this?* •

Task 3 : Make spatial object with the calibration points, their Pb concentrations, and the indicator. Compute and plot the empirical variogram. Model it and fit the model. •

Q3 : *What is your fitted variogram model? Does this indicator have a clear spatial structure?* •

Q4 : *What is the effective range?* •

Q5 : *Why is the total sill so much lower than for the example in the exercise?* •

Task 4 : Make a map of the probability of a `TRUE` indicator over the prediction grid, with the sample indicators over-printed. •

Q6 : *Describe the spatial pattern of the predicted probability.* •

References

[1] P Goovaerts. *Geostatistics for natural resources evaluation.* Applied Geostatistics. Oxford University Press, New York; Oxford, 1997. 1

[2] P Goovaerts, R Webster, and J-P Dubois. Assessing the risk of soil contamination in the Swiss Jura using indicator geostatistics. *Environmental and Ecological Statistics*, 4(1):31–48, 1997. 1

Index of R Concepts

< operator, 2

as.numeric, 3

ceiling, 23
coordinates (package:sp), 3, 17

data.frame, 11
dim, 13

fit.variogram (package:gstat), 6
floor, 23

gstat package, 33

ifelse, 9

lattice package, 33

max, 22, 23
min, 23

order, 12, 13

pmax, 8, 22
pmin, 8
pt, 25, 26

qt, 22
qz, 22

row.names, 13

sp package, 33
spplot (package:sp), 4, 8

variogram (package:gstat), 4, 5
vgm (package:gstat), 6