
Applied geostatistics

Exercise 5: Predicting from point samples (Part 2)

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

June 27, 2014

Contents

1	Kriging weights	1
1.1	Answers	3
2	Block kriging	4
2.1	Theory	5
2.2	Practice	5
2.3	Answers	11
3	Universal kriging	11
3.1	Theory	12
3.2	Looking for a trend	12
3.3	Fitting the residual variogram	16
3.3.1	* Fixing the sill	19
3.4	Predicting with UK	20
3.5	* UK in a neighbourhood	24
4	* Spurious patchiness with kriging in a neighbourhood	27
4.1	Answers	30
5	* Insight into the UK system	32
5.1	Answers	38
6	Self-test	38

Version 2.4. Copyright © 2006–2009 ITC, 2010–2012, 2014 University of Twente 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 (d.g.rossiter@utwente.nl).

References	41
Index of R concepts	42

玉不琢不成器
“Jade that is not chiseled cannot become a gem”
– Chinese proverb

After completing this exercise you should be able to:

1. Understand how Ordinary Kriging (OK) weights are affected by point configuration and variogram model;
2. Predict block averages for attributes using block Ordinary Kriging (BOK);
3. Predict attributes over a region using Universal Kriging (UK).

A supplementary exercise, **ex5a**, continues with Kriging with External Drift (KED), which is mathematically the same as Universal Kriging but uses one or more feature-space predictors to model part of the variability.

1 Kriging weights

In this section we take a short detour from R to investigate the Ordinary Kriging (OK) system in an interactive graphical program. This allows us to see the effect of variogram model, model parameters, sample point distribution, and prediction point location on the predictions and their variances.

For this section you must have access to an MS-Windows system¹. We will use the E{Z}-Kriging program² written by Dennis J. J. Walvoort, Wageningen University, the Netherlands. Although this is (and will likely remain) Version 0.2, it nicely illustrates how the kriging system works.

Note: We have also provided a **spreadsheet** in Microsoft Excel format named **OK_Explained.xls** in folder **datasets/xls** with the course material.

Task 1 : Extract the executable program **E_Z_Kriging.exe** from the “zip” archive file **E_Z_Kriging.zip** to a convenient location on your system. •

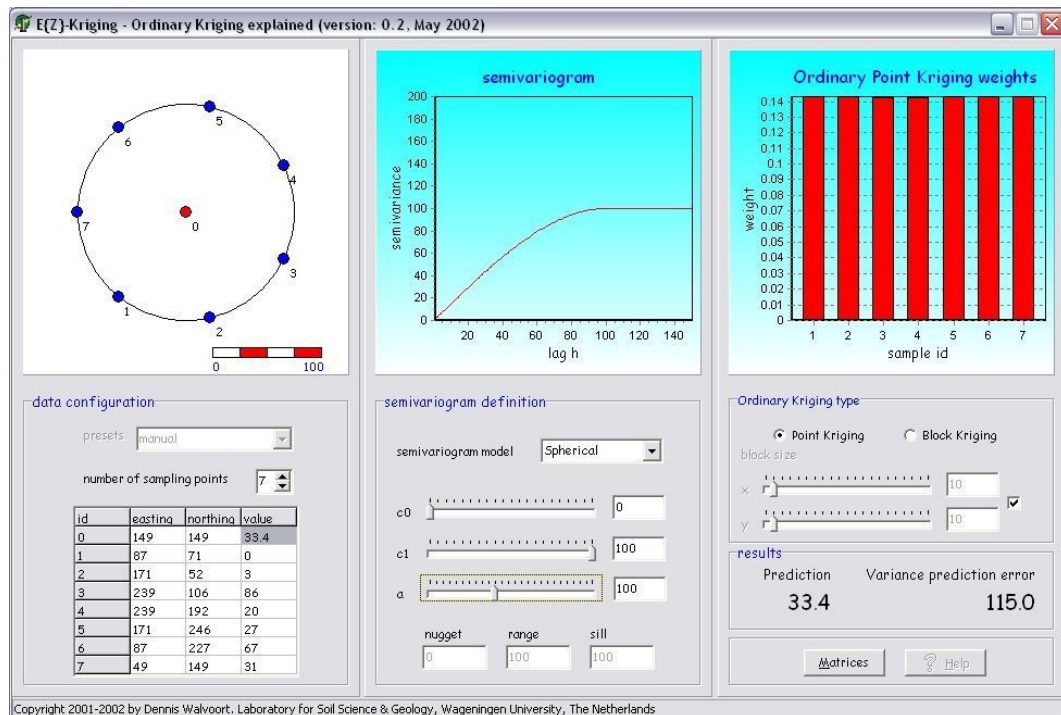
Task 2 : Review sections 1 (“Introduction”) and 2 (“Graphical User Interface”) of the the presentation **E_Z_Kriging.pdf**. This explains how to use the program. •

Task 3 : Run the program **E_Z_Kriging.exe**. •

The default configuration shows one prediction point in the middle of a circle of radius 100 with seven equally-spaced data points, a spherical semi-variogram model with range 100, sill 100, and no nugget. The seven points have various data values.

¹ This can be a virtual system, e.g., Wine, see <http://www.winehq.org/>

² https://wiki.52north.org/bin/view/AI_GEOSTATS/SWEZKriging



Q1 : What is the prediction and its variance?

Jump to A1 •

Q2 : What are the absolute and relative weights of the sample points?

Jump to A2 •

Task 4 : Keeping the point configuration the same, change some data values.

•

Q3 : What happens to the prediction and its variance? Explain. *Jump to A3 •*

Task 5 : Move point 2 next to point 1.

•

Q4 : What are now the weights? Explain.

Jump to A4 •

Task 6 : Move point 2 half-way between point 1 and the prediction point, i.e. so that it 'masks' point 1.

•

Q5 : What are now the weights? Explain.

[Jump to A5](#) •

Task 7 : With the slider, reduce the range from 100 slowly towards 40. •

Q6 : What happens to the weights as the range is reduced? Explain. [Jump to A6](#) •

Task 8 : Move point 2 back to point 1, to form a two-point cluster. •

Q7 : What are now the weights? Explain.

[Jump to A7](#) •

Task 9 : Restore the original configuration (equally-spaced points). With the slider increase and reduce the **range** parameter and observe the predictions and their variances. •

Q8 : What happens to the predictions and their variances as the range is increased? reduced? Explain. [Jump to A8](#) •

Task 10 : Restore the original configuration (equally-spaced points). With the slider increase the **nugget** and observe the predictions and their variances. •

Q9 : What happens to the predictions and their variance as the nugget is increased? Explain. [Jump to A9](#) •

This should give you some ideas on more experimentation; see also the suggestions in section 3 (“Interesting features to explore”) of the presentation `E_Z_Kriging.pdf`.

1.1 Answers

A1 : Prediction 33.4, variance 115 (remember, this is in squared units). [Return to Q1](#) •

A2 : All are equal, $1/7 = 0.142857\dots$ [Return to Q2](#) •

A3 : The prediction changes, but the variance stays the same. This is because

the prediction depends on the data values, but the variance only on the point configuration. [Return to Q3](#) •

A4 : Points 3–7 have weights about 0.16, the weights of points 1 and 2 are reduced to about 0.9 (depending on exact configuration of the two points). Together the two points have only about 0.18 weight, this is the clustering effect. [Return to Q4](#) •

A5 : Point 2 now has about half the total weight (0.5); points 3–7 almost equal (about 0.1 each); point 1 now has a **negative** weight. The high weight for point 2 is explained by its proximity to the prediction point; the negative weight for point 1 because it is masked by point 2, i.e. in the same direction from the prediction point. [Return to Q5](#) •

A6 : They become more equal, until the range reaches the innermost point, at which time they all have equal weight. When all sample points are outside the range there is no longer any spatial dependence with the prediction point. [Return to Q6](#) •

A7 : Points 1 and 2 have reduced weights, each about 0.1 (combined 0.2); the others have equal weights of about 0.15. Even though the range is short, there is still spatial dependence **between sample points** if they are closer than the range, so there is still a clustering effect in that case. [Return to Q7](#) •

A8 : Predictions do not change in this case because of the isotropic point distribution. Increased range decreases the kriging variance, and vice-versa. This is because the data values at sample points are more spatially-correlated with the prediction point at longer ranges. [Return to Q8](#) •

A9 : Predictions do not change, but the kriging variance is increased. There is spatial uncertainty at the sample points as well as at the prediction point. [Return to Q9](#) •

2 Block kriging

This section introduces prediction on **blocks**, i.e. areas larger than the sample support. This is an easy modification of **punctual** OK prediction. The differences are:

1. the prediction is a **block average**, which smooths out local extremes;
2. the short-range variability (within a block) is removed, so the **kriging variances are lower** than for punctual OK.

Block Kriging can be applied to any type of kriging; in this section we apply it to OK.

2.1 Theory

Block kriging make estimates at blocks of a defined size, with unknown mean (which must also be estimated) and no trend; centred at the coördinates specified in the prediction object. Each block B is estimated as the weighted average of the values at all sample points \mathbf{x}_i , as with OK:

$$\hat{Z}(B) = \sum_{i=1}^N \lambda_i z(\mathbf{x}_i) \quad (1)$$

As with OK, the weights λ_i sum to 1, so that the estimator is unbiased. The predictions are on the same support of the data values, that is, they are the average of all supports within the block.

The Block Kriging system is derived as for the OK system, producing these equations:

$$\sum_{j=1}^N \lambda_j \gamma(\mathbf{x}_i, \mathbf{x}_j) + \psi(B) = \bar{\gamma}(\mathbf{x}_i, B), \quad \forall i$$

This equation implies that:

1. The semivariances are averages of the point semivariances between **sample points** and **all** the points in the **block** to be predicted;
2. The semivariance with a block is written as $\bar{\gamma}$, the overline indicating an **average**;
3. The left-hand side (semivariances between sample points) is the same as in OK.

It is impossible to compute an infinite number of semivariances and then average these, so in practice the block to be predicted is **discretized** by a square grid of points.

The key benefit from block ordinary kriging (BK) is the factor by which the estimation variance is reduced:

$$\bar{\gamma}(B, B) = \frac{1}{|B|^2} \int_B \int_B \gamma(\mathbf{x}, \mathbf{x}') d\mathbf{x} d\mathbf{x}' \quad (2)$$

As the block size $|B|$ approaches zero, the double integral also approaches zero; in fact this is the limit. This shows that OK is a special case of BK.

In practice, the above integral is calculated by **discretizing** the block into n points:

$$\bar{\gamma}(B, B) \approx \frac{1}{|B|^2} \sum_{i=1}^n \sum_{j=1}^n w_i w_j \gamma(\mathbf{x}_i, \mathbf{x}_j) \quad (3)$$

where $\sum_i w_i = 1$; the weights are set by their position within the unit block.

2.2 Practice

We continue with the Jura dataset from Exercise 4.

Task 11 : To set up this exercise:

1. If R is not already running, start it. If you haven't already done so, load the `gstat` and `sp` libraries, as shown in the previous exercises.
2. If the calibration dataset `jura.cal` is not loaded as a spatial object, do so. It was saved as part of R data file `JuraEx4.RData` in Exercise 4; this can be loaded into the workspace with the `load` function.
3. If the fitted variogram model `vmf` from Exercise 4 §4.1 is no longer in the workspace, re-create it.
4. If the prediction grid `jura.raster` from Exercise 4 §4.4 is no longer in the workspace, re-create it.
5. If the kriging prediction `k.grid` from Exercise 4 §4.4 is no longer in the workspace, re-create it.

•

If you followed the instructions in Exercise 4, these should all have been saved in file `JuraEx4.RData`, so they can be restored with the `load` function:

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

Note: If this file is not in the current directory, you either have to change the directory with the `setwd` function, or add the full path to the argument of the `load` function.

Block kriging with the `krige` function of `gstat` is exactly like OK, with one additional argument: `block`, which gives the dimensions of the block as a list. For the usual case of a 2D block (surface area), this is a list of two dimensions (which are usually, but not necessarily the same). The list is formed with the `c` function.

Task 12 : Use the fitted variogram model to predict at all the grid points by Ordinary Kriging (OK) with a block size of 50 x 50 m. •

The coördinates are in km, so the desired block size is expressed as 0.05 km.

```
> bk.grid <- krige(Co ~ 1, loc = jura.cal, newdata = jura.raster,
+   model = vmf, block = c(0.05, 0.05))
```

```
[using ordinary kriging]
```

```
> summary(bk.grid)
```

```
Object of class SpatialGridDataFrame
Coordinates:
      min  max
[1,] 0.3 5.10
[2,] 0.4 5.75
Is projected: NA
proj4string : [NA]
Grid attributes:
      cellcentre.offset cellsize cells.dim
1           0.325      0.05         96
```


2	0.425	0.05	107
---	-------	------	-----

Data attributes:

var1.pred	var1.var
Min. : 2.92	Min. : 0.345
1st Qu.: 8.48	1st Qu.: 2.495
Median : 9.73	Median : 3.857
Mean : 9.54	Mean : 6.097
3rd Qu.:10.98	3rd Qu.:10.459
Max. :15.69	Max. :13.223

Q10 : *What are the minimum, mean and maximum predictions? How do these compare with the values for the OK predictions of Exercise 4 §4.4? Explain any differences.* Jump to A10 •

Q11 : *What are the minimum, mean and maximum kriging variances? How do these compare with the for the OK kriging variances of Exercise 4 §4.4? Explain any differences.* Jump to A11 •

We can see the effect of blocking visually by side-by-side plots:

Task 13 : Plot the BK predictions and their variances, next to and with the same scale as the OK predictions and variances. •

To specify a scale to `spplot`, we use the `at` optional argument. We want to use the same scale, so we first compute the extremes of both prediction grids together, once for the predictions and once for their variances, with the `range` function:

```
> range(k.grid$var1.pred, bk.grid$var1.pred)
[1] 2.9057 15.8122

> range(k.grid$var1.var, bk.grid$var1.var)
[1] 0.3455 15.0050
```

We round these a bit down and up, and break at each integer; we can use the `:` simple sequence operator for the integers, otherwise the `seq` function:

```
> (at.pred <- 2:16)
[1] 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

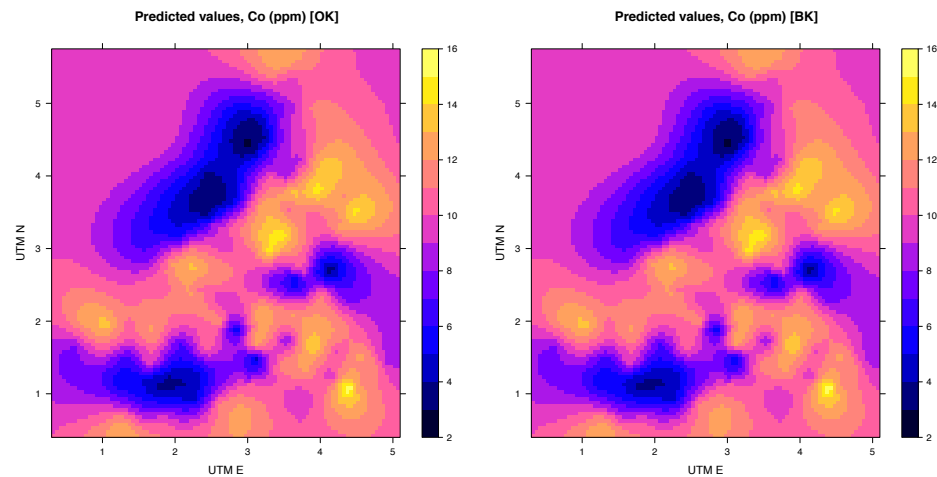
> (at.var <- seq(0, 15.5, by = 0.5))
[1] 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0
[14] 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0 10.5 11.0 11.5 12.0 12.5
[27] 13.0 13.5 14.0 14.5 15.0 15.5
```

To plot several graphs together, we save each one as a Trellis graphics object, and then use the `split` and more optional arguments to the `print` function

of the `lattice` package. (See [4, §5.4] for more on how to print several graphs in one window, or open multiple windows.)

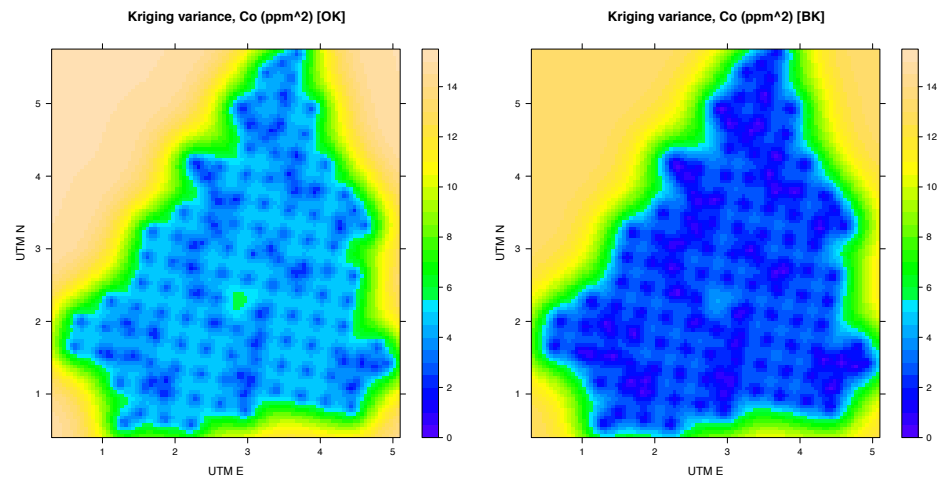
First the two predictions:

```
> plot.1 <- spplot(k.grid, zcol="var1.pred",
+                 col.regions=bpy.colors(64),
+                 main="Predicted values, Co (ppm) [OK]",
+                 xlab="UTM E", ylab="UTM N",
+                 scales=list(draw=T), at=at.pred)
> #
> plot.2 <- spplot(bk.grid, zcol="var1.pred",
+                 col.regions=bpy.colors(64),
+                 main="Predicted values, Co (ppm) [BK]",
+                 xlab="UTM E", ylab="UTM N",
+                 scales=list(draw=T), at=at.pred)
> #
> print(plot.1, split=c(1, 1, 2, 1), more=T)
> print(plot.2, split=c(2, 1, 2, 1), more=F)
```



Now the two kriging variances:

```
> plot.1 <- spplot(k.grid, zcol="var1.var",
+                 col.regions=topo.colors(64),
+                 main="Kriging variance, Co (ppm^2) [OK]",
+                 xlab="UTM E", ylab="UTM N",
+                 scales=list(draw=T),
+                 at=seq(0, 15.5, by=0.5))
> #
> plot.2 <- spplot(bk.grid, zcol="var1.var",
+                 col.regions=topo.colors(64),
+                 main="Kriging variance, Co (ppm^2) [BK]",
+                 xlab="UTM E", ylab="UTM N",
+                 scales=list(draw=T),
+                 at=seq(0, 15.5, by=0.5))
> #
> print(plot.1, split=c(1, 1, 2, 1), more=T)
> print(plot.2, split=c(2, 1, 2, 1), more=F)
```



We can also compare predictions and variances by computing their differences.

Task 14 : Compute and plot a difference map between BK and OK predictions and variances. •

First the computation: we make a new data frame, with two fields, which we explicitly name **pred** and **var**. For easier interpretation of the numbers we round the differences to a meaningful precision, consistent with the original observations.

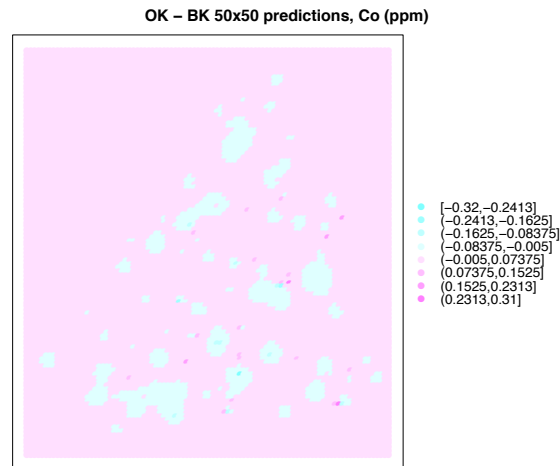
```
> diff.ok.bk <- data.frame(
+   pred = round(k.grid$var1.pred - bk.grid$var1.pred, 2),
+   var = round(k.grid$var1.var - bk.grid$var1.var, 3))
> summary(diff.ok.bk)
```

pred		var	
Min. :	-0.32000	Min. :	1.31
1st Qu. :	0.00000	1st Qu. :	1.76
Median :	0.00000	Median :	1.77
Mean :	0.00001	Mean :	1.77
3rd Qu. :	0.00000	3rd Qu. :	1.78
Max. :	0.31000	Max. :	1.78

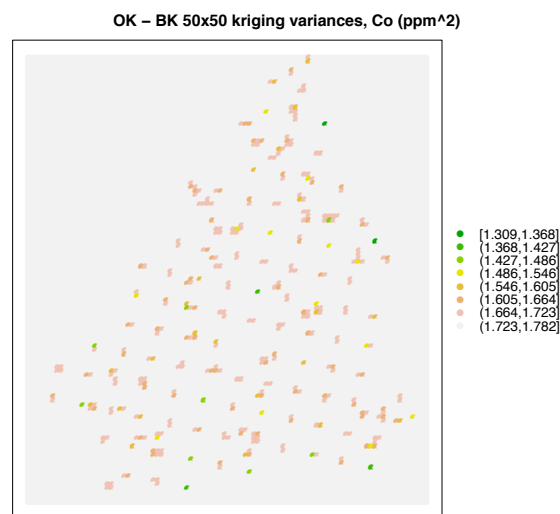
```
> coordinates(diff.ok.bk) <- coordinates(bk.grid)
```

Then we plot the differences:

```
> print(spplot(diff.ok.bk, zcol="pred", cuts=8,
+   col.regions=cm.colors(8),
+   main="OK - BK 50x50 predictions, Co (ppm)",
+   key.space="right"))
```



```
> print(spplot(diff.ok.bk, zcol="var", cuts=8,
+           col.regions=terrain.colors(8),
+           main="OK - BK 50x50 kriging variances, Co (ppm^2)",
+           key.space="right"))
```



Q12 : *Are there any differences in pattern or values of the predictions and prediction variances between OK and BK?* Jump to A12 •

Task 15 : Clean up the workspace from this section. •

We keep the interpolation grid `jura.raster`, the OK and BK predictions `k.grid` and `bk.grid`, and the omnidirectional variogram model `vmf` for future use.

```
> rm(at.pred, at.var, plot.1, plot.2)
```

2.3 Answers

A10 : Predicted values are almost identical.

[Return to Q10](#) •

A11 : Kriging variances are substantially lower: the minimum is 0.346 for BK, 1.847 for OK. The mean is 6.097 instead of 7.863. This reduction represents the variance within each block and the reduced variance between sample points and the prediction block.

[Return to Q11](#) •

A12 : For the predicted values the two maps appear identical; the difference map shows very small absolute differences (both positive and negative). For the variances all values are much lower with BK, but with less difference near observation points and especially near point clusters.

[Return to Q12](#) •

3 Universal kriging

Universal Kriging (UK) models both a **regional trend** and a **local spatial dependence** together. The trend is expressed as a function of the **coordinates**. The idea is that some of observed variation is due to a trend, and the variation in what is left over (the **residuals**) can be explained as a random field.

The key point of UK is that it can account for random fields that are *not* 1st-order stationary; that is, the expected value can change smoothly over the field. Of course, even if the field is indeed 1st-order stationary the observed values will vary over the field, with some spatial correlation structure; but in that case the expected value (before observation) is the same everywhere. If there is clear evidence that this is not true, e.g., a clear regional trend which can be explained by physical or geographical principles, UK can be used to implicitly remove the trend.

Note: Some examples of non-stationary processes are (1) the elevation of an aquifer above some datum, when the rock stratum containing the aquifer is tilted; (2) mean annual rainfall over a region which ranges from coast to inland.

Note: The term “Universal Kriging” is used by some authors also to include a trend in one or more **feature-space** predictors, i.e. co-variables. In this course we call this **Kriging with External Drift** (KED); the mathematics are the same but the predictors are not all in geographic space.

The steps in UK are:

1. Compute the regional **trend**;
2. Compute and model the variogram of the **residuals** from the regional trend;

3. Use this variogram model and the trend to **predict**.

In UK the trend is implicit in the kriging equations; this is useful if we want to restrict the kriging neighbourhood.

3.1 Theory

The trend is modelled as a **linear combination** of p known **base functions** $f_j(s)$ and p unknown constants β_j ; these are the **parameters** of the base functions:

$$Z(\mathbf{x}_i) = \sum_{j=1}^p \beta_j f_j(\mathbf{x}_i) + e(\mathbf{x}_i) \quad (4)$$

The base functions for **linear** drift (a plane) are:

$$f_0(\mathbf{x}) = 1, f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = x_2 \quad (5)$$

where x_1 is one coördinate (say, E) and x_2 the other (say, N), and $f_0(\mathbf{x}) = 1$ estimates the global mean (as in OK).

For **full quadratic** drift (a possibly elongated bowl or hump) or we also include second-order terms:

$$f_3(\mathbf{x}) = x_1^2, f_4(\mathbf{x}) = x_1 x_2, f_5(\mathbf{x}) = x_2^2 \quad (6)$$

A point is predicted as in OK:

$$\hat{Z}(\mathbf{x}_0) = \sum_{i=1}^N \lambda_i z(\mathbf{x}_i) \quad (7)$$

However, in UK the weights λ_i for each sample point take into account both the global trend (drift, violation of first-order stationarity) and local effects. The unbiasedness condition is expressed with respect to the trend as well as the overall mean (as in OK):

$$\sum_{i=1}^N \lambda_i f_k(\mathbf{x}_i) = f_k(\mathbf{x}_0), \quad \forall k \quad (8)$$

That is, the expected value at each point of all the base functions must be that predicted by that function. The first of these is the overall mean (as in OK).

3.2 Looking for a trend

The first step, then, is to see if there is any regional trend. The R^2 of the trend gives some idea of the potential improvement in predictions.

For this section we continue with the Jura data set, specifically the calibration set `jura.cal`. We keep working with the target variable Cobalt concentration (attribute `Co`). This does not show a dramatic regional trend, so

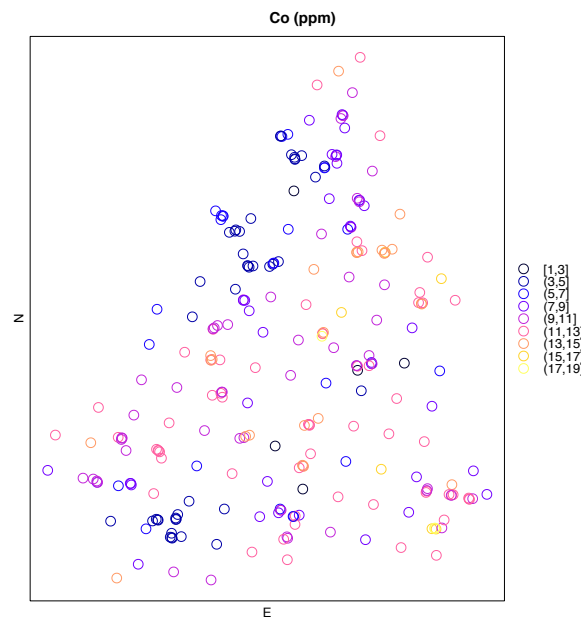
that UK does not differ too much from OK. Still, it illustrates the principles and procedures.

Note: In this analysis we will use **ordinary least squares** (OLS) to compute the trend. This is only approximately correct; since we have evidence of spatial dependence between values, we should use **generalized least squares** (GLS), weighting the points according to the **covariance structure**. But this leads to a “chicken-and-egg” problem: we don’t know the covariance structure of the residuals until we compute them, and for that we need the trend, and for that we should know the covariance structure, etc. In practice, OLS trend surfaces are used as a first approximation to the GLS surface, and the spatial structure of the residuals is modelled from this. We examined this issue in Exercise 3, optional §5.

Task 16 : Visualize the regional trend in Co concentration. •

We use `spplot` with a colour ramp:

```
> print(spplot(jura.cal, zcol="Co", col.regions=bpy.colors(64),
+             cuts=seq(1,19,by=2), key.space="right",
+             main="Co (ppm)", xlab="E", ylab="N",
+             pch=1, cex=1.5))
```



Q13 : Does there appear to be any regional trend in Co concentration? (Hint: is there a trend from blue to yellow, via purple and orange?) In what direction? How strong is it? Does it appear to be a plane or a higher-order surface? Jump to A13 •

Another way to visualize this is to compute a trend surface on the prediction grid using the `krige` of the `gstat` package without a model, as we did in Exercise 3 §2. However, before we can do this, we must ensure that the names of the coördinates within the prediction grid are the same in the

prediction and sample datasets. Otherwise, the formula $Co \sim X + Y$ used in the `krige` function will not find the named coordinates in the prediction grid.

As it stands, the `jura.raster` object, of class `SpatialGrid`, does not have any coordinate names, as we can see with the `str` function on the `@bbox` slot:

```
> str(jura.raster@bbox)

num [1:2, 1:2] 0.3 0.4 5.1 5.75
- attr(*, "dimnames")=List of 2
..$ : chr [1:2] "X" "Y"
..$ : chr [1:2] "min" "max"

> str(jura.cal@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"
```

We can specify the names with the `coordnames` function:

```
> coordnames(jura.raster) <- c("X", "Y")
> str(jura.raster@bbox)

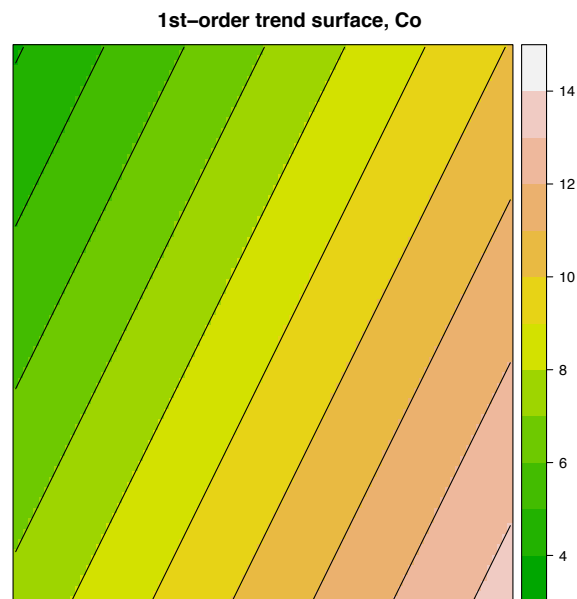
num [1:2, 1:2] 0.3 0.4 5.1 5.75
- attr(*, "dimnames")=List of 2
..$ : chr [1:2] "X" "Y"
..$ : chr [1:2] "min" "max"
```

Now we can compute and display the trend surface:

```
> ts1 <- krige(Co ~ X + Y, loc=jura.cal,
+             newdata=jura.raster, model=NULL)

[ordinary or weighted least squares prediction]

> print(spplot(ts1, zcol="var1.pred", contour=T,
+             pretty=T, col.regions=terrain.colors(64),
+             main="1st-order trend surface, Co"))
```

Q14 : What is the approximate azimuth (direction) of this first-order trend surface? *Jump to A14* •

Task 17 : Compute the regional trend. •

```
> (lm.co.xy <- lm(Co ~ coordinates(jura.cal), data = jura.cal))
```

Call:

```
lm(formula = Co ~ coordinates(jura.cal), data = jura.cal)
```

Coefficients:

(Intercept)	coordinates(jura.cal)X
7.142	1.297
coordinates(jura.cal)Y	
-0.639	

```
> summary(lm.co.xy)
```

Call:

```
lm(formula = Co ~ coordinates(jura.cal), data = jura.cal)
```

Residuals:

Min	1Q	Median	3Q	Max
-9.200	-2.706	0.193	2.535	7.782

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	7.142	0.732	9.75	< 2e-16 ***
coordinates(jura.cal)X	1.297	0.217	5.98	7.5e-09 ***
coordinates(jura.cal)Y	-0.639	0.163	-3.92	0.00011 ***

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

Residual standard error: 3.32 on 256 degrees of freedom
 Multiple R-squared: 0.145, Adjusted R-squared: 0.139
 F-statistic: 21.7 on 2 and 256 DF, p-value: 1.9e-09

Q15 : *How much of the variation in Co concentration is explained by the trend?* *Jump to A15 •*

Task 18 : Compute the direction of the trend. •

The two slope coefficients in E and N determine the **azimuth** (angle from N): the plane increases towards the angle determined by the ratio between them. The **atan2** function, in the form **atan2(y, x)**, returns the angle (in radians) between the “x”-axis (as defined in the call) and the vector from the origin to “(x, y)” (also as defined in the call). For the azimuth, we want angle from N, so we call this in the form **atan2(E, N)** where E and N are extracted from the model fit with the **coefficients** function. This must be converted to degrees by the ratio $180/\pi$.

```
> coefficients(lm.co.xy)

      (Intercept) coordinates(jura.cal)X coordinates(jura.cal)Y
              7.1423                1.2968                 -0.6393

> atan2(coefficients(lm.co.xy)[2], coefficients(lm.co.xy)[3]) *
+   (180/pi)

coordinates(jura.cal)X
                116.24
```

Q16 : *In which direction does the plane increase?* *Jump to A16 •*

3.3 Fitting the residual variogram

The next step is to compute the **residual variogram**. The **variogram** function of the **gstat** package computes the trend and, from that, the residuals, from the **model formula**. Recall: for OK we used the model formula $\text{Co} \sim 1$, meaning to model Co from itself only (local spatial dependence). Now we use the formula $\text{Co} \sim \text{coordinates(jura.cal)}$ to model the trend.

Note: This variogram is not strictly correct, since it is computed from the OLS residuals. Since the residuals have not yet been computed, it's impossible to determine their spatial structure, which should be used to correctly estimate the trend parameters using generalized least squares (GLS). If the observation points are approximately regularly spaced this is not a problem. But in the general case, the proper approach is to use residual maximum likelihood (REML) to estimate trend and covariance parameters together; see Lark et al. [3]. Here we continue with the naïve approach, since (1) it is much easier to explain; (2) the observations are mostly well-spaced.

Task 19 : Compute the residual variogram of Co concentration, taking into account a first-order regional trend; compare it with the original variogram (no trend) both numerically and graphially. •

We will want to compare this variogram to that computed in Exercise 3, §3.3, so we re-compute that one; recall we used a cutoff of 1.6 km.

```
> v <- variogram(Co ~ 1, loc = jura.cal, cutoff = 1.6)
> vr <- variogram(Co ~ coordinates(jura.cal), loc = jura.cal,
+   cutoff = 1.6)
```

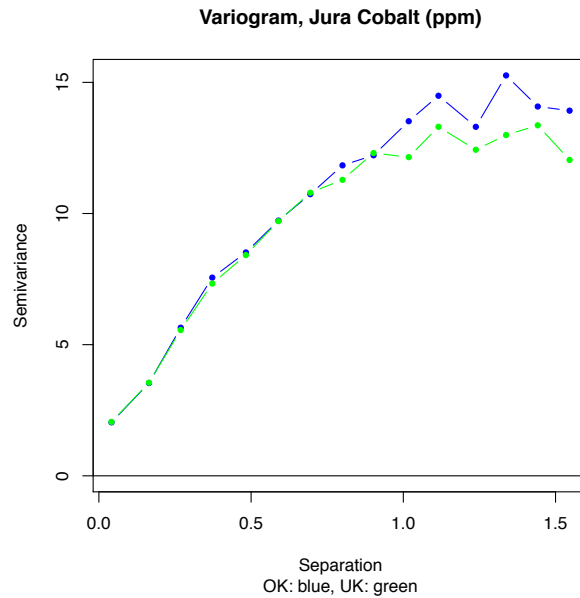
These have the same structure, the same distances, and the same numbers of point-pairs in each bin. So we can build one frame to compare them:

```
> (compare.v <- data.frame(np = v$np, separation = v$dist,
+   gamma.ok = v$gamma, gamma.uk = vr$gamma, gamma.dif = v$gamma -
+   vr$gamma))
```

	np	separation	gamma.ok	gamma.uk	gamma.dif
1	281	0.041378	2.0379	2.0569	-0.019034
2	209	0.164607	3.5373	3.5518	-0.014464
3	401	0.268811	5.6470	5.5585	0.088480
4	656	0.372567	7.5569	7.3306	0.226280
5	744	0.482890	8.5160	8.4170	0.098967
6	543	0.589803	9.7310	9.7087	0.022311
7	784	0.694563	10.7370	10.7946	-0.057611
8	966	0.800257	11.8357	11.2860	0.549710
9	749	0.902205	12.2191	12.3063	-0.087253
10	1115	1.017696	13.5181	12.1483	1.369825
11	1243	1.115531	14.4904	13.3057	1.184770
12	1170	1.238103	13.3056	12.4321	0.873472
13	1171	1.337529	15.2638	12.9918	2.272037
14	1266	1.441534	14.0793	13.3621	0.717197
15	1334	1.546090	13.9222	12.0425	1.879742

These can be plotted as two variograms on one graph, using the generic `plot` function to set up the plot and display the first variogram, and then using the `points` function to add the second variogram to the plot. Note the use of the `type` optional argument with the value of "b" to plot **both** the points and connecting lines; also the `ylim` optional argument to display the variogram from $\gamma = 0$, to visualize the nugget effect.

```
> plot(compare.v$gamma.ok ~ compare.v$separation, pch=20,
+   col="blue", type="b",
+   xlab="Separation", ylab="Semivariance",
+   ylim=c(0, max(compare.v$gamma.ok, compare.v$gamma.uk)),
+   main="Variogram, Jura Cobalt (ppm)",
+   sub="OK: blue, UK: green")
> points(compare.v$gamma.uk ~ compare.v$separation, pch=20,
+   col="green", type="b")
> abline(h=0, lty=1)
```



Q17 : What are the major differences between the original and residual variograms? Why do they not differ very much? *Jump to A17 •*

The next step is to model the residual variogram.

Task 20 : Select a variogram model and fit parameters for the residual variogram. •

In the absence of any evidence to the contrary, we should use the same model form as for the original variogram, i.e. pentaspherical in this case. We first fit by eye and then automatically. We already know the fitted model for OK; it is a good starting point, with slightly reduced sill and range.

```
> vmf

      model  psill  range
1  Nug  1.3712 0.0000
2  Pen 12.9322 1.5239

> vrm <- vgm(12, "Pen", 1.2, 1.4)
> plot.1 <- plot(vr, pl=T, model=vrm,
+               main="Jura Co; residuals from 1st-order trend",
+               sub="Eyeball model")
> (vrmf <- fit.variogram(vr, vrm))

      model  psill  range
1  Nug  1.3845 0.0000
2  Pen 11.4658 1.3469

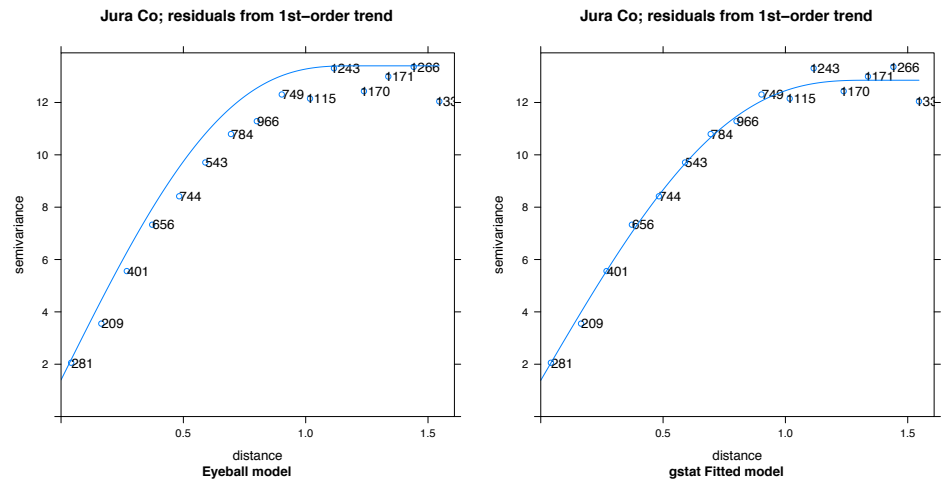
> attributes(vrmf)$SSErr

[1] 3173.6
```

```

> plot.2 <- plot(vr, pl=T, model=vrmf,
+               main="Jura Co; residuals from 1st-order trend",
+               sub="gstat Fitted model")
> print(plot.1, split=c(1,1,2,1), more=T)
> print(plot.2, split=c(2,1,2,1), more=F)

```



We compare the two parts of the model (nugget and structure); the first reported result is the difference in the nugget parameters and the second in the structural model parameters. The range difference of the nugget is of course zero.

```

> vmf$range - vrmf$range

[1] 0.00000 0.17703

> vmf$psill - vrmf$psill

[1] -0.013331 1.466441

```

Q18: What are the differences between the original and residual variogram?

Jump to A18 •

3.3.1 * Fixing the sill

This last difference reveals a problem: the nugget of the fitted residual variogram model is not the same as that for the original variogram model.

```

> vrmf$psill[1]

[1] 1.3845

> vmf$psill[1]

[1] 1.3712

> (vrmf$psill[1] - vmf$psill[1])/vmf$psill[1]

[1] 0.009722

```

In this case the difference is quite small with respect to the total sill (about 1%), but still, in theory removing a long-range trend should have no effect on the nugget.

One way to handle this problem, well-justified by theory, is to fix the nugget from the original variogram surface and not allow it to change, specifying the optional `fit.sills` argument to the `fit.variogram` function, to fix the first sill (nugget) and allow the second (structural) to change. We take the nugget from the fitted model of original values, extracted from the fitted model object:

```
> (vrmf.2 <- fit.variogram(vr, vgm(12, "Pen", 1.2, vmf[1,
+   "psill"]), fit.sills = c(F, T)))

  model  psill  range
1  Nug   1.3712 0.0000
2  Pen  11.4670 1.3406

> attributes(vrmf.2)$SSErr

[1] 3193.6
```

Note that the nugget remains the same as specified from the original model. Comparing to the residual variogram fit without this restriction:

```
> print(vrmf)

  model  psill  range
1  Nug   1.3845 0.0000
2  Pen  11.4658 1.3469

> attributes(vrmf)$SSErr

[1] 3173.6

> rm(vrmf.2)
```

In this case, fixing the nugget slightly increases the structural sill and decreases the range; these are minor adjustments which would have little practical consequences. However, this technique is useful if the automatically-fitted nugget of a residual variogram is unrealistically low (see the self-test for an example).

3.4 Predicting with UK

Now we are in a position to predict with UK.

Task 21 : Predict the Co concentration over the region using UK. •

The `krige` command is almost the same as for OK; we just need to specify the trend and the changed variogram model.

Note: The coördinate names of the prediction grid and the data frame with the sample points must be the same; we did that at the beginning of §3.

Finally we can compute the UK predictions; note the use of the formula $X + Y$ to specify the trend; these variables refer both to the samples `jura.cal` and the prediction points `jura.raster`.

```
> uk.grid <- krige(Co ~ X + Y, loc = jura.cal, newdata = jura.raster,
+   model = vrmf)

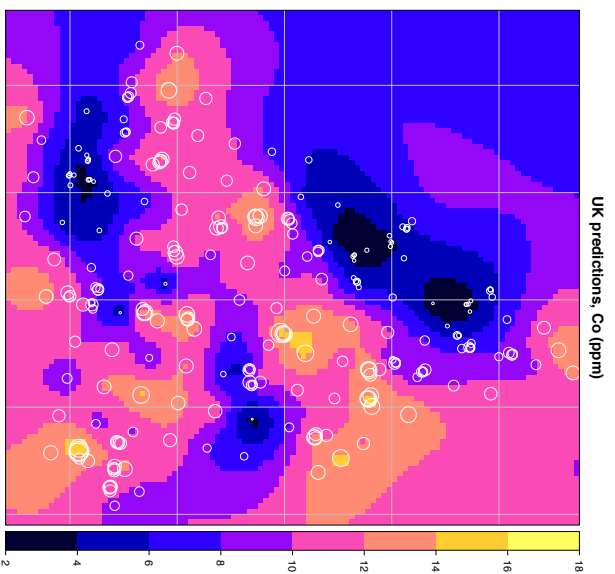
[using universal kriging]

> summary(uk.grid)

Object of class SpatialGridDataFrame
Coordinates:
  min  max
X 0.3 5.10
Y 0.4 5.75
Is projected: NA
proj4string : [NA]
Grid attributes:
  cellcentre.offset cellsize cells.dim
X           0.325      0.05         96
Y           0.425      0.05        107
Data attributes:
  var1.pred      var1.var
Min.   : 2.93   Min.   : 1.86
1st Qu.: 7.55   1st Qu.: 4.30
Median : 9.38   Median : 5.70
Mean   : 9.20   Mean   : 8.28
3rd Qu.:11.01   3rd Qu.:12.74
Max.   :15.82   Max.   :19.85
```

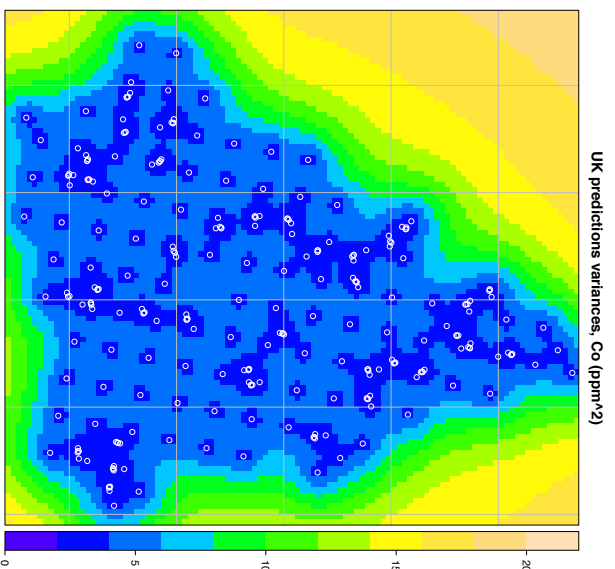
We then plot the predictions:

```
> print(spplot(uk.grid, zcol="var1.pred",
+   col.regions=bpy.colors(64),
+   pretty=T, cuts=8,
+   main="UK predictions, Co (ppm)",
+   key.space="right",
+   panel=function(x,y,z, ...) {
+     panel.levelplot(x, y, z, ...)
+     panel.grid(h=-1,v=-1, col="gray", lty=1)
+     panel.points(coordinates(jura.cal),
+       cex=3*jura.cal$Co/max(jura.cal$Co),
+       pch=1, col="white")
+   })
```



And then we plot the prediction variances:

```
> print(splot(uk.grid, zcol="var1.var",
+           col.regions=topo.colors(64),
+           pretty=T, cuts=8,
+           main="UK predictions variances, Co (ppm^2)",
+           key.space="right",
+           panel=function(x,y,z, ...) {
+             panel.levelplot(x, y, z, ...)
+             panel.grid(h=-1,v=-1, col="gray", lty=1)
+             panel.points(coordinates(jura.cal),
+                             pch=1, col="white")
+           }
+           ))
```



Task 22 : Compute and plot a difference map between UK and OK predictions and variances. •

First the computation: we make a new data frame, with two fields, which we explicitly name `pred` and `var`:

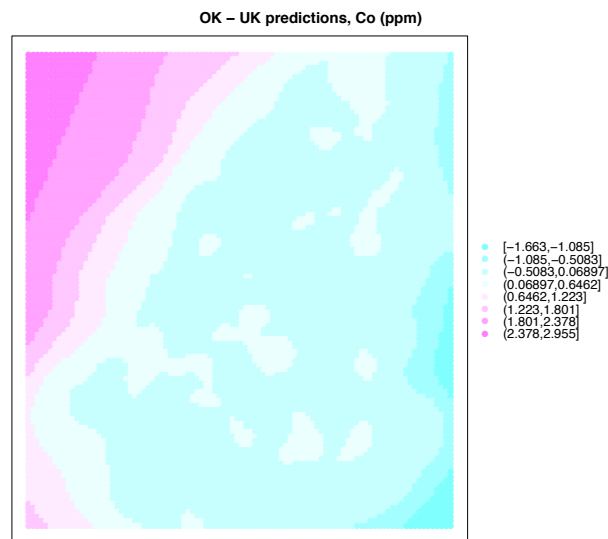
```
> diff.ok.uk <- data.frame(pred = k.grid$var1.pred - uk.grid$var1.pred,
+   var = k.grid$var1.var - uk.grid$var1.var)
> str(diff.ok.uk)

'data.frame':      10272 obs. of  2 variables:
 $ pred: num  2.96 2.92 2.88 2.85 2.81 ...
 $ var : num -4.84 -4.72 -4.59 -4.47 -4.35 ...

> coordinates(diff.ok.uk) <- coordinates(uk.grid)
```

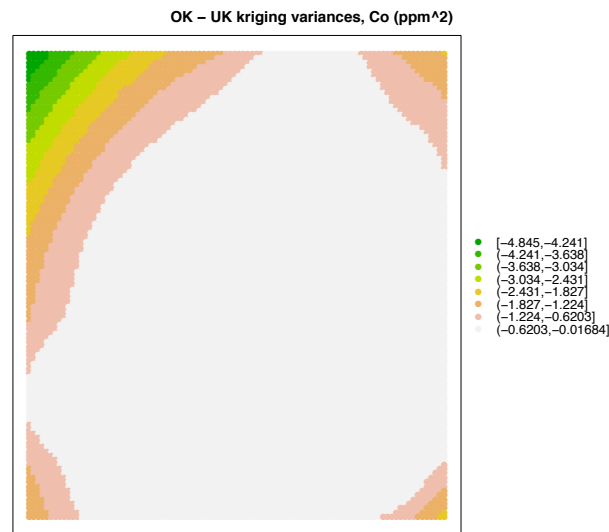
Then we plot these differences; first the predictions:

```
> print(spplot(diff.ok.uk, zcol="pred", pretty=T, cuts=8,
+   col.regions=cm.colors(64),
+   main="OK - UK predictions, Co (ppm)",
+   key.space="right"))
```



Then the variances:

```
> print(spplot(diff.ok.uk, zcol="var", pretty=T, cuts=8,
+   col.regions=terrain.colors(64),
+   main="OK - UK kriging variances, Co (ppm^2)",
+   key.space="right"))
```



Q19 : What are the major differences? Explain.

Jump to A19 •

Task 23 : Clean up from this section. •

```
> rm(diff.ok.uk)
```

3.5 * UK in a neighbourhood

In this **optional** section we see how to restrict the UK system to a local **neighbourhood**. This allows local fitting of the trend surface, i.e. a non-stationary trend. There is no theory for the selection of the neighbourhood radius, but it often works well in practice if the radius is approximately the **variogram range**, in this case about 1.3 km. Although the resulting map is less smooth than the unrestricted UK map, the predictions are generally quite similar, because the observation points within the range of the residual process receive almost all of the kriging weight.

Task 24 : Compute UK predictions with a trend neighbourhood of 1.3 km; compare with the UK predictions with a regional trend. •

The only modifications to the `krige` function call are the addition of two optional arguments:

1. `maxdist`: the maximum distance from a prediction point to fit the trend surface; this restricts the neighbourhood;
2. `nmin`: the **minimum number of observations** to use to make a prediction; this ensures that a trend surface is not fitted with too few points. We chose a minimum of 16 observations.

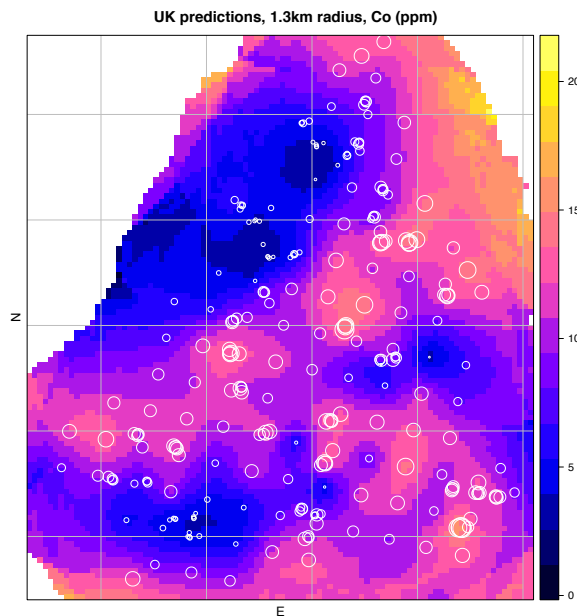
```

> uk13.grid <- krige(Co ~ X + Y, loc = jura.cal,
+                   newdata = jura.raster, model = vrmf,
+                   maxdist=1.3, nmin=16)

[using universal kriging]

> print(spplot(uk13.grid, zcol="var1.pred",
+             col.regions=bpy.colors(64),
+             main="UK predictions, 1.3km radius, Co (ppm)",
+             xlab="E", ylab="N",
+             panel=function(x,y,z, ...) {
+               panel.levelplot(x, y, z, ...)
+               panel.grid(h=-1,v=-1, col="gray", lty=1)
+               panel.points(coordinates(jura.cal),
+                             cex=3*jura.cal$Co/max(jura.cal$Co),
+                             pch=1, col="white")
+             })))

```



Q20 : What are the major differences between UK over the whole region and with a restricted neighbourhood? Explain. [Jump to A20](#) •

To help answer this, we summarize and plot the prediction differences:

```

> uk13.grid$pred.diff <- uk13.grid$var1.pred - uk.grid$var1.pred
> summary(uk13.grid$pred.diff)

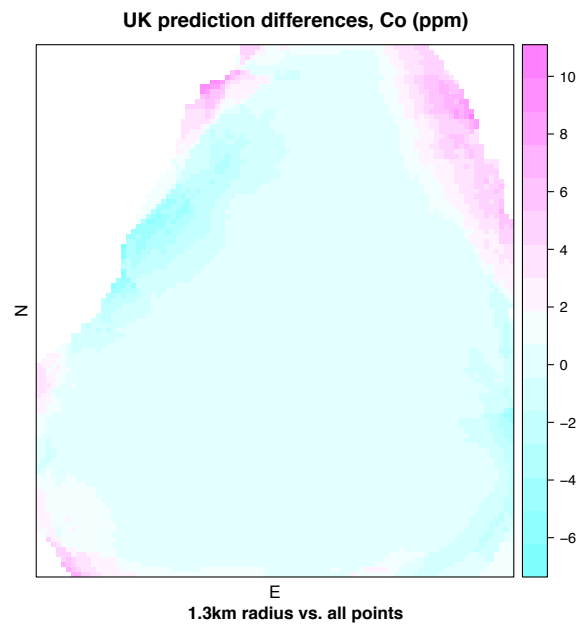
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
-6.23	-0.07	0.01	0.24	0.14	9.96	1720

```

> print(spplot(uk13.grid, zcol="pred.diff",
+             col.regions=cm.colors(64),
+             main="UK prediction differences, Co (ppm)",
+             sub="1.3km radius vs.\ all points",
+             xlab="E", ylab="N"))

```

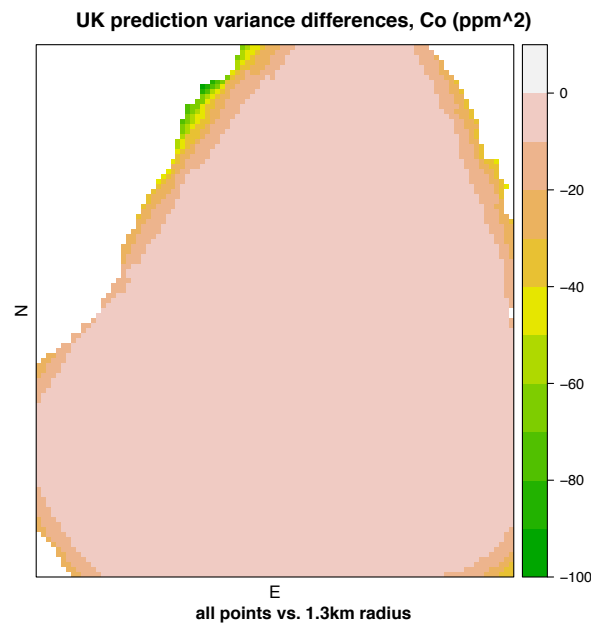


And we see how the predication variances differ:

```
> uk13.grid$var.diff <- uk.grid$var1.var - uk13.grid$var1.var
> summary(uk13.grid$var.diff)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
-91.00	-1.64	-0.01	-2.84	0.00	0.00	1720

```
> print(spplot(uk13.grid, zcol="var.diff",
+             pretty=T, cuts=8,
+             col.regions=terrain.colors(64),
+             main="UK prediction variance differences, Co (ppm^2)",
+             sub="all points vs. 1.3km radius",
+             xlab="E", ylab="N")))
```



Q21 : What are the major differences between the prediction variances for UK over the whole region and with a restricted neighbourhood? Explain.

[Jump to A21](#) •

Challenge: Repeat this section with different neighbourhoods, and explore the effect of changing neighbourhood size on the predictions and variances.

4 * Spurious patchiness with kriging in a neighbourhood

This optional section shows that using a neighbourhood for kriging can introduce spurious patchiness in the result. This effect increases with increasing nugget variance as a proportion of the total sill. To illustrate this, we create a pure nugget variogram model for Co with the same total sill as the fitted model, and then use it to predict by OK (1) using all points and (2) in a neighbourhood.

Q22 : In theory, what should be the spatial pattern of the predictions and their variances if kriging with a pure nugget variogram model? What should be the predicted value?

[Jump to A22](#) •

```
> (vmn <- vgm(0, "Pen", vmf[2, "range"], sum(vmf[, "psill"])))

      model psill range
1   Nug 14.303 0.0000
2   Pen  0.000 1.5239

> ok.n <- krige(Co ~ 1, loc=jura.cal,
+              newdata=jura.raster, model=vmn)

[using ordinary kriging]
```

```
> summary(ok.n@data)

      var1.pred      var1.var
Min.   :9.3    Min.   :14.4
1st Qu.:9.3    1st Qu.:14.4
Median :9.3    Median :14.4
Mean   :9.3    Mean   :14.4
3rd Qu.:9.3    3rd Qu.:14.4
Max.   :9.3    Max.   :14.4

> ok.n.w <- krige(Co ~ 1, loc=jura.cal,
+               newdata=jura.raster, model=vmn,
+               maxdist=1.3, nmin=16)
```

[using ordinary kriging]

```
> summary(ok.n.w@data)

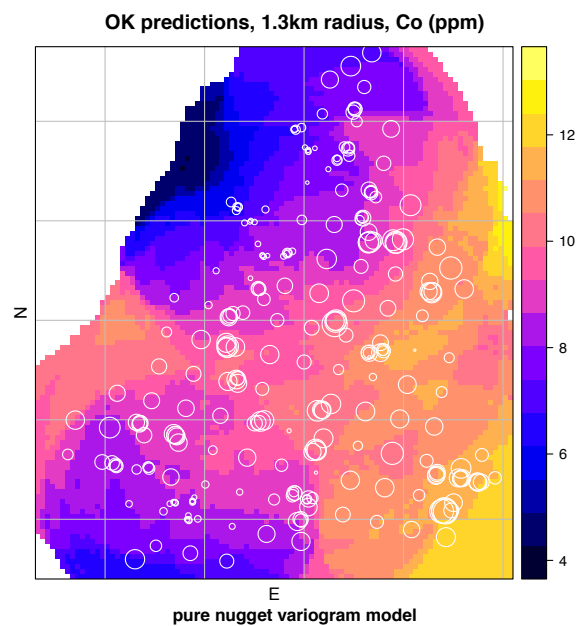
      var1.pred      var1.var
Min.   : 4.26    Min.   :14.4
1st Qu.: 8.08    1st Qu.:14.5
Median : 9.28    Median :14.6
Mean   : 9.21    Mean   :14.6
3rd Qu.:10.52    3rd Qu.:14.7
Max.   :13.04    Max.   :15.2
NA's   :1720     NA's   :1720
```

```
> mean(jura.cal$Co)
```

```
[1] 9.3009
```

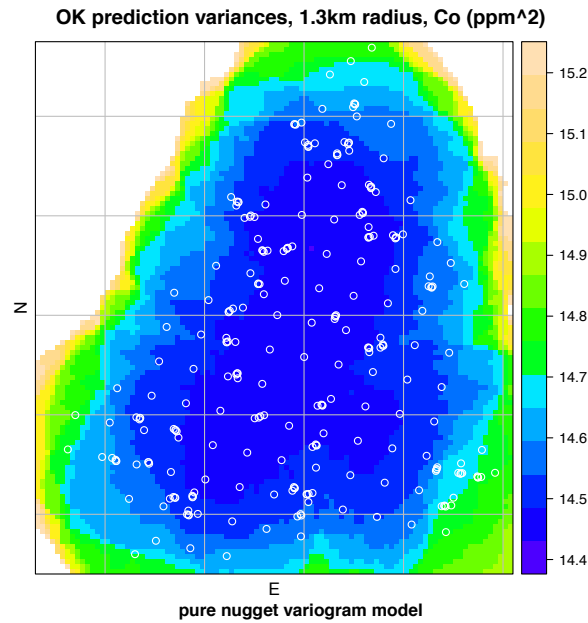
Here is a map of the OK predictions:

```
> print(spplot(ok.n.w, zcol="var1.pred",
+             col.regions=bpy.colors(64),
+             main="OK predictions, 1.3km radius, Co (ppm)",
+             sub="pure nugget variogram model",
+             xlab="E", ylab="N",
+             panel=function(x,y,z, ...) {
+               panel.levelplot(x, y, z, ...)
+               panel.grid(h=-1,v=-1, col="gray", lty=1)
+               panel.points(coordinates(jura.cal),
+                             cex=3*jura.cal$Co/max(jura.cal$Co),
+                             pch=1, col="white")
+             })))
```



And this is a map of the OK prediction variances:

```
> print(spplot(ok.n.w, zcol="var1.var",
+             col.regions=topo.colors(64),
+             main="OK prediction variances, 1.3km radius, Co (ppm^2)",
+             sub="pure nugget variogram model",
+             xlab="E", ylab="N",
+             panel=function(x,y,z, ...) {
+               panel.levelplot(x, y, z, ...)
+               panel.grid(h=-1,v=-1, col="gray", lty=1)
+               panel.points(coordinates(jura.cal),
+                             pch=1, col="white")
+             })))
```



Q23 : How do these maps differ from the map that would be produced by using all observations for kriging from a pure nugget variogram model?

[Jump to A23](#) •

Q24 : Which prediction is correct, the one produced by OK with all observations or that by OK in a neighbourhood?

[Jump to A24](#)

•

Challenge: Repeat this section but with the fitted variogram model for Co. Is there any spurious patchiness? Compare with the results of this section. (Hint: consider the nugget's proportion of the total sill.)

4.1 Answers

A13 : Yes; the lower values appear to be concentrated on the NW side and the higher on the SE. The trend appears to run to the ENE (increasing); a plane seems sufficient to model it.

[Return to Q13](#) •

A14 : The increase is towards the ESE, approximately 120° from North. [Return to Q14](#) •

A15 : 13.9% is explained (see the adjusted R^2).

[Return to Q15](#) •

A16 : Azimuth 116°.

[Return to Q16](#) •

A17 : They are quite similar up till about 1 km separation; then the residual variogram is significantly lower. This means that the trend has taken out some of the long-range structure. The short-range dependence is almost the same. The range of the residual variogram also appears to be a bit shorter. The small difference reflects the low R^2 of the trend surface. [Return to Q17 •](#)

A18 : The residual variogram model has somewhat lower sill (about 1.5 lower) and shorter range (about 0.18 km shorter). This reflects the variability taken out by the trend. The nuggets are almost equal. [Return to Q18 •](#)

A19 : The OK predictions are higher in the NW corner and lower in the SE corner than the UK predictions. This is because the global trend increases from NW to SE.

The OK kriging variances are much lower than for UK in the corners away from sample points. This is because there is no error from the trend surface, which error increases out from the centroid (centre of mass) of the sample points from which the surface was calculated. [Return to Q19 •](#)

A20 : The UK with restricted neighbourhood is patchy; the slight regional trend can not be seen; instead the trend surface conforms locally to observations in the neighbourhood. The blank areas are those without at least 16 observations within 1.3 km. The differences are most marked near the edges, with few observations, where the few nearby observations result in a local trend. [Return to Q20 •](#)

A21 : The UK with restricted neighbourhood never has lower prediction variance, because it is using less information. In the centre of the area the difference is effectively zero, but at the edges, where fewer points are available within the radius, the uncertainty increases dramatically. Recall, the grid cells on the edges are predicted from just 16 observations. [Return to Q21 •](#)

A22 : The prediction and variance should be everywhere the same, because there is no spatial dependence, hence no extra information from being nearer to known observations. The predicted value is everywhere the arithmetic mean of all observations – this is at the same time the spatial mean, because there is no spatial dependence to consider. The prediction variance is everywhere the total sill of the variogram model. [Return to Q22 •](#)

A23 : The predicted values vary greatly depending on the neighbourhood. Instead of the single value 9.76 there are predictions over a wide range, from 4.265 to 13.042. OK with a pure nugget variogram model in a neighbourhood is essentially a moving average. [Return to Q23 •](#)

A24 : If the empirical variogram shows no evidence of spatial dependence, i.e., if a pure nugget variogram model is the best fit, OK over a neighbourhood shows spurious patchiness and false “precision”. The correct prediction is everywhere the mean, with a single prediction variance. [Return to Q24 •](#)

5 * Insight into the UK system

In this **optional** section we look at how **gstat** solves the universal kriging system. This is an extension of what we saw for the OK system in the previous exercise.

Because the size of the kriging matrix depends on the number of sample points, we first reduce the size of the problem.

Task 25 : Select the first six sample points of the prediction sample, and their Co concentration. Define a one-point **SpatialPoints** object at (3.0,2.5) to be predicted.

```
> (jura.cal.6 <- jura.cal[1:6, "Co"])

      coordinates      Co
1 (2.386, 3.077)  9.32
2 (2.544, 1.972) 10.00
3 (2.807, 3.347) 10.60
4 (4.308, 1.933) 11.92
5 (4.383, 1.081) 16.32
6 (3.244, 4.519)  3.50

> (jura.pt <- SpatialPoints(data.frame(X = 3, Y = 2.5)))

SpatialPoints:
      X      Y
[1,] 3 2.5
Coordinate Reference System (CRS) arguments: NA
```

Task 26 : Predict at this point with UK with a first-order trend, showing the kriging system, by using the optional **debug.level** argument to the **krige** function.

Note that the **residual** variogram model **vrmf** must be used for Universal Kriging.

The optional **debug.level** argument is passed by **krige** to the **predict.gstat** function which is what is actually doing the kriging, see **?predict.gstat** for various useful values. With **debug.level=32** this prints the covariance matrices, design matrices, solutions, and kriging weights.

```
> k.pt <- krige(Co ~ X + Y, locations = jura.cal.6, newdata = jura.pt,
+              model = vrmf, debug.level = 32)

[using universal kriging]
we're at location X: 3 Y: 2.5 Z: 0
zero block size
we're at point X: 3 Y: 2.5 Z: 0

# X:
```

```

Matrix: 6 by 3
row 0:      1      2.386      3.077
row 1:      1      2.544      1.972
row 2:      1      2.807      3.347
row 3:      1      4.308      1.933
row 4:      1      4.383      1.081
row 5:      1      3.244      4.519
[using generalized covariances: max_val - semivariance()]
# Covariances (x_i, x_j) matrix C (lower triangle only):
Matrix: 6 by 6
row 0:  12.8502702      0      0      0      0
      0      0
row 1:  0.126071034  12.8502702      0      0
      0      0
row 2:  4.18617845      0      12.8502702      0
      0      0
row 3:      0      0      0      12.8502702
      0      0
row 4:      0      0      0      0      1.03997138
      12.8502702      0
row 5:      0      0      0.00984774609      0
      0      12.8502702

# X'C-1 X:
Matrix: 3 by 3
row 0:  0.415790555      1.37795277      1.09602191
row 1:  1.37795277      4.83623914      3.4492001
row 2:  1.09602191      3.4492001      3.45686012

# beta:
Vector: dim: 3
      15.894892      0.599556878      -2.91373745
# Cov(beta), (X'C-1 X)-1:
Matrix: 3 by 3
row 0:  96.9795145      -19.7729994      -11.0188553
row 1:  -19.7729994      4.74849477      1.53119647
row 2:  -11.0188553      1.53119647      2.25508225

# Corr(beta):
Matrix: 3 by 3
row 0:      1      -0.921413494      -0.745101238
row 1:  -0.921413494      1      0.467920145
row 2:  -0.745101238      0.467920145      1

# X0 (X values at prediction location x0):
Matrix: 3 by 1
row 0:      1
row 1:      3
row 2:      2.5

# BLUE(mu), E(y(x0)) = X0'beta:
Vector: dim: 1
      10.4092191
# Covariances (x_i, x_0), C0:
Matrix: 6 by 1

```

```

row 0:      1.11375274
row 1:      2.16156596
row 2:      0.965328826
row 3:      0
row 4:      0
row 5:      0

# C-1 C0:
Matrix: 6 by 1
row 0:      0.0677451658
row 1:      0.167547081
row 2:      0.05305226
row 3:      0
row 4:      0
row 5: -4.06563582e-05

# [a] Cov_ij(B,B) or Cov_ij(0,0):
Matrix: 1 by 1
row 0:      12.8502702

# [c] (x0-X'C-1 c0)'(X'C-1 X)-1(x0-X'C-1 c0):
Matrix: 1 by 1
row 0:      1.30739838

# [b] c0'C-1 c0:
Matrix: 1 by 1
row 0:      0.488828306

# Best Linear Unbiased Predictor:
Vector: dim: 1
          10.3409813
# MSPE ([a]-[b]+[c]):
Matrix: 1 by 1
row 0:      13.6688402

# kriging weights:
Matrix: 6 by 1
row 0:      0.199175674
row 1:      0.353321362
row 2:      0.147214249
row 3:      0.0970322758
row 4:      0.118795406
row 5:      0.0844610338

```

Working through this output, we see:

1. The **prediction point** and **block size** (here, zero);
2. The **design matrix** X with three columns: (1) a column of 1's to predict the spatial mean; (2) a column of the X coordinates of the sample points; (3) a column of the Y coordinates of the sample points;
3. The **covariance matrix** C between sample points; this depends of course on the **variogram model**;
4. The **quadratic form** $X^T C^{-1} X$ used in many of the subsequent calcula-

- tions; this accounts for the covariance between sample points;
5. The **trend coefficients** β , estimated by GLS using the covariance matrix; note that this replaces the **spatial mean** from the OK system; there is a coefficient for the intercept and one each for each coordinate;
 6. The **covariance** of the trend coefficients;
 7. The **correlation matrix** of the trend coefficients;
 8. The values of the **basis functions** at the prediction points, i.e. 1 and the actual coordinates of the point;
 9. The **BLUE** of the trend at the prediction point;
 10. The **covariance vector** C_o between the prediction point and each sample point;
 11. The product $C^{-1}C_o$;
 12. The **within-block** or **at-point** covariance, here just the total variogram sill (estimating the overall variance);
 13. A matrix product used in the BLUP;
 14. A matrix product used in the BLUP;
 15. The **BLUP** at the prediction point; this is the **kriging prediction**;
 16. The **kriging prediction variance** at the prediction point;
 17. The **kriging weights**, i.e. the weights given to each sample point when their values are summed into the BLUP.

Q25 : *In which direction is the trend?*

Jump to A25 •

The final result is in the kriging object:

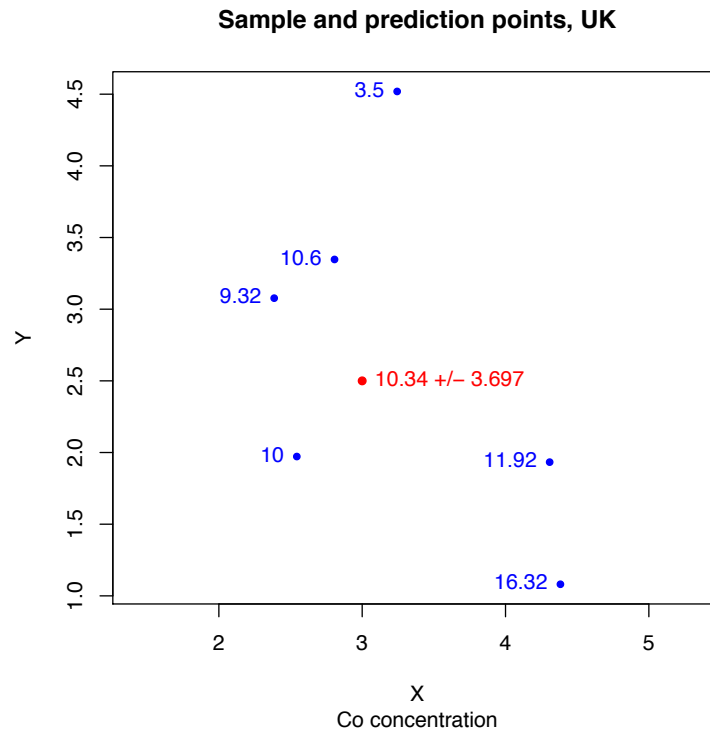
```
> print(k.pt)

coordinates var1.pred var1.var
1      (3, 2.5)    10.341    13.669
```

Task 27 : Plot the sample points and their Co concentration, with the prediction point and its predicted concentration and standard error of the prediction. •

```
> plot(coordinates(jura.cal.6), col="blue",
+       pch=20, asp=1,
+       main="Sample and prediction points, UK",
+       sub="Co concentration")
> text(coordinates(jura.cal.6)[,"X"],
+       coordinates(jura.cal.6)[,"Y"],
+       col="blue", pos=2, jura.cal.6$Co)
> points(coordinates(jura.pt), col="red",
+        cex=1.2, pch=20)
```

```
> text(coordinates(jura.pt), col="red", pos=4,
+       paste(round(k.pt$var1.pred,2), "+/-",
+             round(sqrt(k.pt$var1.var),3)))
```



Note that the OK prediction (previous exercise) with this same system was 10.08 ± 3.764 . The extra information on the trend in the neighbourhood of these six points has reduced the prediction variance somewhat, and the trend itself has changed the prediction; it is slightly higher because the point is a bit towards the SSE from the centroid of the six known points, which have slightly higher values.

Task 28 : Display this plot but with the kriging weights, rather than the predictions. Also display the OK weights for comparison. •

To find the weights, we capture the output into a workspace variable with `capture.output`, and then search through it for the weights.

```
> tmp.uk <- capture.output(krige(Co ~ X + Y, locations = jura.cal.6,
+   newdata = jura.pt, model = vrmf, debug.level = 32))
> tmp.ok <- capture.output(krige(Co ~ 1, locations = jura.cal.6,
+   newdata = jura.pt, model = vmf, debug.level = 32))

> str(tmp.uk)

chr [1:108] "[using universal kriging]" ...

> (ix <- which(tmp.uk == "# kriging weights:"))

[1] 97
```

```

> (n <- as.numeric(strsplit(tmp.uk[ix + 1], " ")[[1]][2]))

[1] 6

> uk.wt <- NULL
> for (i in 1:n) uk.wt[i] <- as.numeric(strsplit(tmp.uk[ix +
+ 1 + i], ":")[[1]][2])
> print(uk.wt)

[1] 0.199176 0.353321 0.147214 0.097032 0.118795 0.084461

> (ix <- which(tmp.ok == "# kriging weights:"))

[1] 89

> ok.wt <- NULL
> for (i in 1:n) ok.wt[i] <- as.numeric(strsplit(tmp.ok[ix +
+ 1 + i], ":")[[1]][2])
> print(ok.wt)

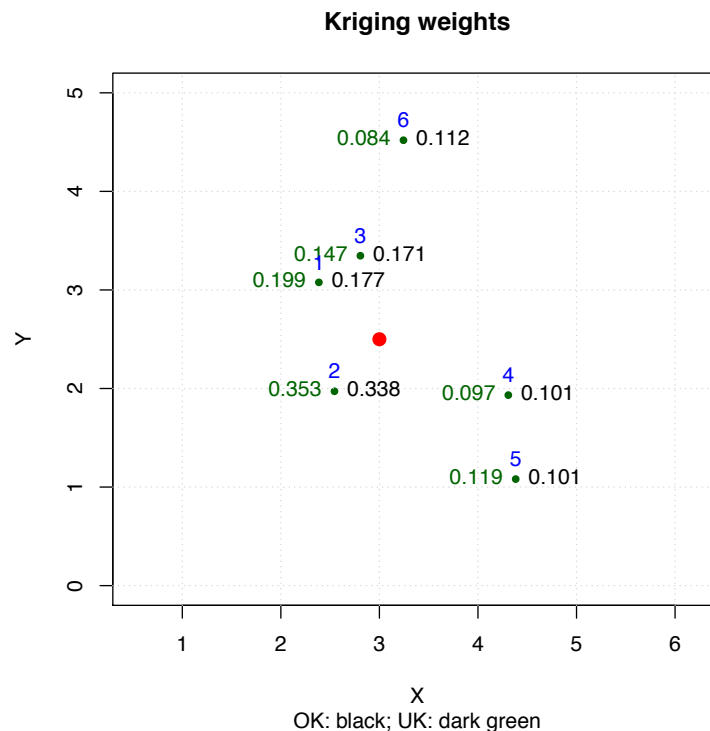
[1] 0.17672 0.33760 0.17108 0.10144 0.10078 0.11239

> print(diff.wt <- round(uk.wt - ok.wt, 4))

[1] 0.0225 0.0157 -0.0239 -0.0044 0.0180 -0.0279

> plot(coordinates(jura.cal.6), col="darkgreen",
+ pch=20, asp=1, ylim=c(0,5),
+ main="Kriging weights",
+ sub="OK: black; UK: dark green")
> grid()
> text(coordinates(jura.cal.6)[,"X"],
+ coordinates(jura.cal.6)[,"Y"],
+ col="darkgreen", pos=2, round(uk.wt, 3))
> text(coordinates(jura.cal.6)[,"X"],
+ coordinates(jura.cal.6)[,"Y"],
+ col="black", pos=4, round(ok.wt, 3))
> text(coordinates(jura.cal.6)[,"X"],
+ coordinates(jura.cal.6)[,"Y"],
+ col="blue", pos=3, 1:6)
> points(coordinates(jura.pt), col="red",
+ cex=2, pch=20)

```



Q26 : Which kriging weights changed the most from OK to UK? Explain (hint: look at the previous answer). [Jump to A26](#) •

Task 29 : Clean up from this section. •

```
> rm(ix, n, uk.wt, ok.wt, diff.wt)
> rm(jura.cal.6, jura.pt, k.pt)
```

5.1 Answers

A25 : The trend is the plane defined by the six sample points, taking into account the spatial correlation. It is given by the β coefficients: $z \approx 15.9 + 0.6X - 2.9Y$; the trend is increasing to the SSE. [Return to Q25](#) •

A26 : The largest increase in weight in the UK system is for point 1; point 2 also has a large increase. Points 3 and 6 have the largest decreases, i.e., their OK weights are higher than their UK weights. An interesting change is in points 4 and 5: in OK they get the same weight but in UK the one that is more to the SSE (point 5) has an increased weight, at the expense of point 4. [Return to Q26](#) •

6 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.

For this exercise we use a small dataset of soil samples from Oxfordshire, UK, supplied as the example dataset `oxford` in the `gstat` package; this was originally collected by Burrough et al. [2] and is also used as an example in the text of Burrough and McDonnell [1].

Task 1 : Load the Oxford dataset into the workspace, examine its structure, and convert it to a spatial object by specifying the coordinates. •

Task 2 : Convert the `oxford` object from `SpatialPointsDataFrame` to `SpatialPixelsDataFrame` by specifying that the points are on a grid, with the `gridded` spatial method. •

Task 3 : Display a post-plot of attribute `CEC1` (cation exchange capacity of the topsoil). •

Q1 : *Does there appear to be a trend in the CEC across the study area? If so, in which direction?* •

Task 4 : **Optional:** Compute first- and second-order trend surfaces as linear models and summarize them. •

Q2 : **Optional:** *How much of the variability is explained by the surfaces?* •

Q3 : **Optional:** *What is the azimuth of the direction of maximum increase for the first-order surface?* •

Task 5 : **Optional:** Display these as contour plots. •

Task 6 : Compute the ordinary variogram and the residual variogram from the suspected trends, and plot them on the same variogram graph. •

Q4 : *What are the differences between the ordinary and residual variograms?* •

Task 7 : Model the ordinary and residual variograms with the **same model form** but different fitted parameters. •

Q5 : *What are the parameters of the fitted ordinary and residual variograms? By what proportion does the trend surface reduce the range and sill?* •

Task 8 : Predict the CEC by Block Ordinary and Block Universal Kriging, on 100 x 100 m blocks centred at the sample points. Hint: you can use the same object as both `location` and `newdata`. •

Task 9 : Display the prediction maps with the same colour scale. •

Q6 : *Describe and explain the major spatial differences in the predictions.* •

Task 10 : **Optional:** Display the prediction variances with the same colour scale. •

Q7 : **Optional:** *Describe and explain the major spatial differences in the prediction variances.* •

Task 11 : Clean up the workspace. •

References

- [1] P A Burrough and R A McDonnell. *Principles of geographical information systems*. Oxford University Press, Oxford, 1998. 39
- [2] P A Burrough, P H T Beckett, and M G Jarvis. The relation between cost & utility in soil survey (I-III). *Journal of Soil Science*, 22(3):359–394, 1971. 39
- [3] R. M. Lark, B. R. Cullis, and S. J. Welham. On spatial prediction of soil properties in the presence of a spatial trend: the empirical best linear unbiased predictor (E-BLUP) with REML. *European Journal of Soil Science*, 57(6):787–799, 2006. 16
- [4] D G Rossiter. *Introduction to the R Project for Statistical Computing for use at ITC*. International Institute for Geo-information Science & Earth Observation (ITC), Enschede (NL), 3.7 edition, 2009. URL http://www.itc.nl/personal/rossiter/teach/R/RIntro_ITC.pdf. 8

Index of R Concepts

: operator, 7

at argument (spplot function), 7

atan2, 16

block argument (krige function), 6

c, 6

capture.output, 36

coefficients, 16

coordnames (sp package), 14

debug.level argument (krige function), 32

fit.sills argument (fit.variogram function), 20

fit.variogram (gstat package), 20

gstat package, 5, 6, 14, 16, 31, 39

krige (gstat package), 6, 14, 20, 24, 32

lattice package, 7

load, 6

maxdist argument (krige function), 24

more argument (print function), 7

nmin argument (krige function), 24

oxford dataset, 39

plot, 17

points, 17

predict.gstat (gstat package), 32

print (lattice package), 7

range, 7

round, 9

seq, 7

setwd, 6

sp package, 5

SpatialGrid (sp class), 14

SpatialPoints (sp package), 32

split argument (print function), 7

spplot (sp package), 7, 13

str, 14

type argument (plot function), 17

variogram (gstat package), 16

ylim argument (plot function), 17