Applied geostatistics
Exercise 5b: Predicting from point samples (Part 4)
Lognormal Kriging

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

June 26, 2010

Contents

1 Introduction **1**

2 Lognormal kriging **1**
2.1 Setup 1
2.2 Evaluating (log)normality 2
2.3 Variogram modelling 7
2.4 Ordinary Kriging 8

3 Back-transformation **10**

4 Clean up **16**

5 Answers **16**

References **18**

Index of R concepts **19**

Version 1.1. Copyright © 2008, 2010 University of Twente, Faculty 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).

1 Introduction

Nothing in the formulation of kriging has made any assumptions about the feature-space distribution of the variable to be kriged; the only assumptions are about spatial stationarity. However, the kriging estimator is a weighted average (sum) of nearby values. A distribution that is not symmetric may present problems. For example, in a **positively-skewed** distribution (often found in earth sciences) the few high values will overwhelm all the others, both in variogram estimation and prediction. Theoretically this may be correct but in practice, with small sample sizes, this can lead to over-prediction as well as empirical variograms that are difficult to model.

Therefore, non-symmetric distributions are often **transformed** prior to variogram analysis and kriging. The problem with transformations for kriging in geographic space (as opposed to transformation for linear modelling in feature space) is that we may wants to **back-transform** not only the prediction, but also the prediction variance, to the original measurement scale.

In this short exercise we will examine **lognormal kriging**. The mathematical formulations were derived by Journel [3] and are also presented by Webster and Oliver [4] and Goovaerts [2].

After completing this exercise you should be able to:

1. **transform** a skewed variable to its logarithm;
2. **model** the transformed variable's spatial dependence, and predict at unsampled locations by Ordinary Kriging;
3. **back-transform** the predictions.

2 Lognormal kriging

2.1 Setup

We continue with the Jura dataset from Exercise 4.

Task 1 : 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. Note: it was saved as part of R data file `JuraEx4.RData` in Exercise 4; this can be loaded into the workspace with the `load` method.
3. If the prediction grid `jura.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` method:

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

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

2.2 Evaluating (log)normality

Lognormal kriging is often used when the target variable is strongly right-skewed.

Task 2 : Display histograms of the metals' distribution. •

First we recall which fields represent metal concentrations:

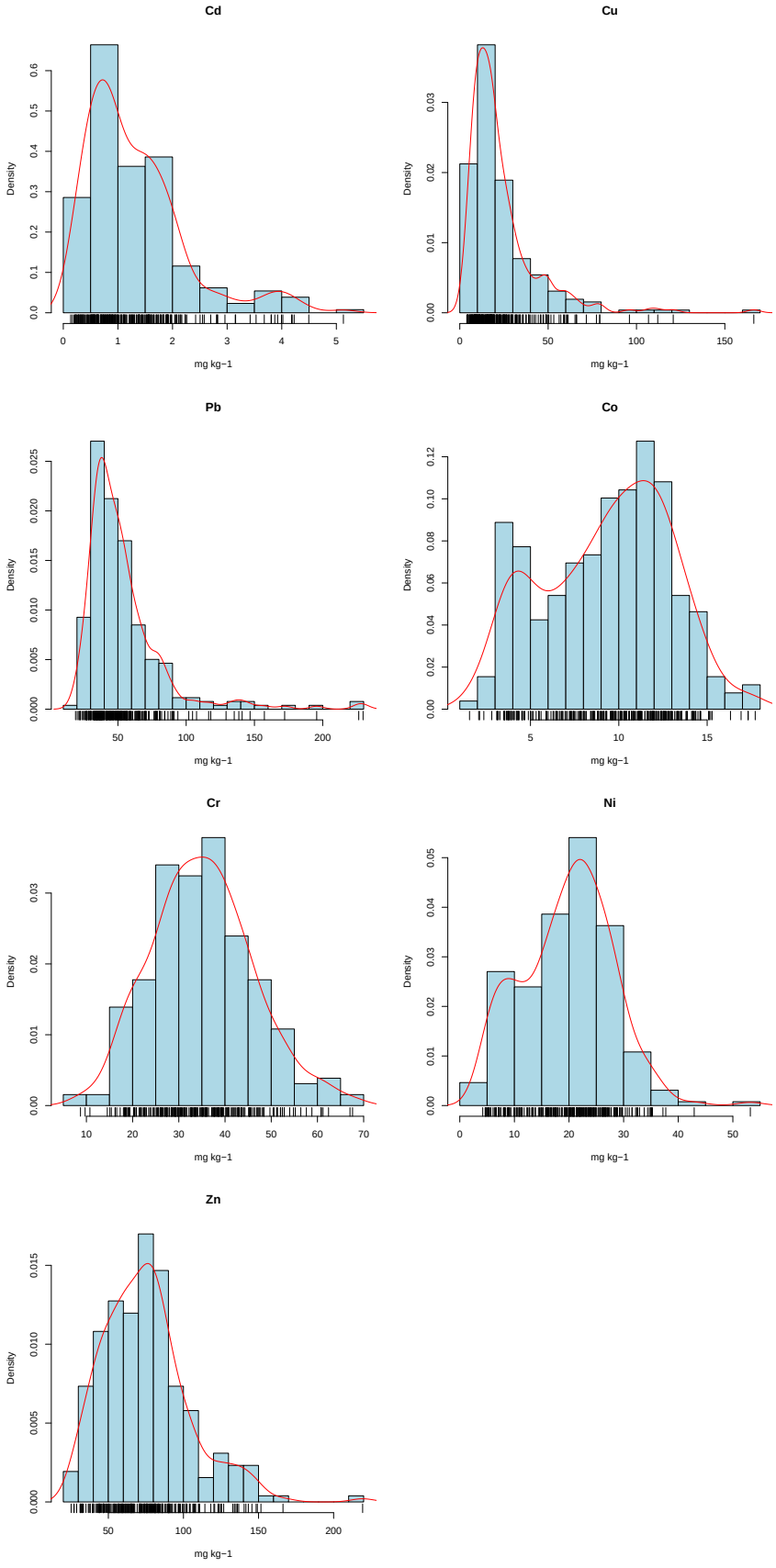
```
> str(jura.cal@data)
```

```
'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  1.74 1.33 1.61 2.15 1.56 ...
 $ Cu  : num  25.72 24.76 8.88 22.7 34.32 ...
 $ Pb  : num  77.4 77.9 30.8 56.4 66.4 ...
 $ Co  : num  9.32 10 10.6 11.92 16.32 ...
 $ Cr  : num  38.3 40.2 47 43.5 38.5 ...
 $ Ni  : num  21.3 29.7 21.4 29.7 26.2 ...
 $ Zn  : num  92.6 73.6 64.8 90 88.4 ...
```

Q1 : *Which fields represent metals?* *Jump to A1* •

Now we display their seven histograms on a single plot (set up as a 4 row, 2 column grid), with rug and kernal densities superimposed:

```
> par(mfrow = c(4, 2))
> for (i in 3:9) {
+     hist(jura.cal@data[, i], freq = FALSE, breaks = 16,
+         col = "lightblue", main = names(jura.cal@data[i]),
+         xlab = "mg kg-1")
+     rug(jura.cal@data[, i])
+     lines(density(jura.cal@data[, i]), col = "red")
+ }
> par(mfrow = c(1, 1))
```


Cd
Density
mg kg−1
Cu
Density
mg kg−1
Pb
Density
mg kg−1
Co
Density
mg kg−1
Cr
Density
mg kg−1
Ni
Density
mg kg−1
Zn
Density
mg kg−1

Q2 : *Which of the metals have a strongly right-skewed distribution, as estimated graphically?* *Jump to A2* •

We can confirm this numerically. Package `e1071` has `skew` and `kurtosis` functions; however it's simple enough to write our own function.

Task 3 : Write a function to compute the skewness of a numeric vector. •

Recall that skewness of a sample is defined as its third standardized moment:

$$g(\mathbf{x}) = (\frac{m_3}{m_2})^{3/2} \tag{1}$$

where m_3 and m_2 are the third and second unstandardized moments, defined as average deviations, to some power, of the n observations, from the sample mean $\bar{x}$:

$$m_k = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^k \tag{2}$$

For a centrally-distributed variable the skew is zero; the cubed differences preserve the sign, so that skew may be positive or negative.

Numerically, the third standardized moment is most easily calculated from the second and third moments about the origin, m'_2 and m'_3 [1, p. 62]:

$$m_3 = m'_3 - 3m'_2\bar{x} + 2\bar{x}^3 \tag{3}$$

$$m'_3 = \frac{1}{n}\sum_{i=1}^{n}x_i^3 \tag{4}$$

$$m'_2 = \frac{1}{n}\sum_{i=1}^{n}x_i^2 \tag{5}$$

The second standardized moment is just the sample variance.

Putting this together, we can write the function:

```
> skew <- function(v) {
+     n <- length(v)
+     mp3 <- sum(v^3)/n
+     mp2 <- sum(v^2)/n
+     mu <- mean(v)
+     m3 <- mp3 - 3 * mp2 * mu + 2 * mu^3
+     return(m3/sd(v)^3)
+ }
```

Note that `sd(v)^3` is just a shorter way to write `var(v)^(3/2)`. This function could have been written more compactly, but here you can see all the steps more clearly.

Task 4 : Apply this function to the metals. •

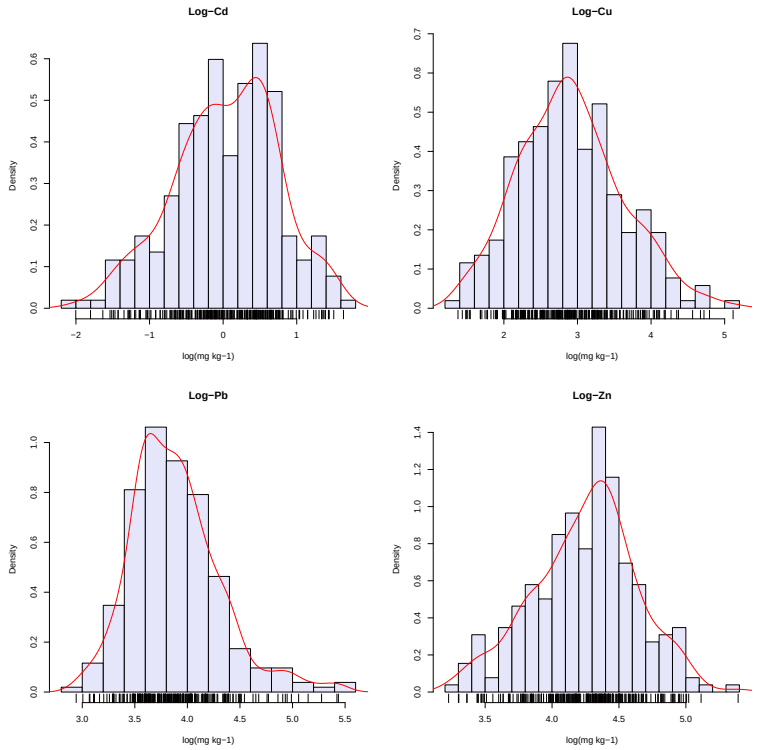
```
> for (i in 3:9) {
+     print(paste(names(jura.cal@data[i]), ":", round(skew(jura.cal@data[,
+         i]), 3)))
+ }

[1] "Cd : 1.499"
[1] "Cu : 2.843"
[1] "Pb : 2.874"
[1] "Co : -0.176"
[1] "Cr : 0.287"
[1] "Ni : 0.158"
[1] "Zn : 1.022"
```

Q3 : *Which of the metals have a strongly right-skewed distribution, as estimated numerically?* *Jump to A3* •

Now we see whether a log-transformation results in an approximately normally-distributed variable for the right-skewed variables `Cd`, `Cu`, `Pb`, and `Zn`.

Task 5 : Display the histograms of log-transformations of the selected variables. •

```
> par(mfrow = c(2, 2))
> for (i in c(3:5, 9)) {
+     hist(log(jura.cal@data[, i]), freq = FALSE, breaks = 16,
+         col = "lavender", main = paste("Log-", names(jura.cal@data[i]),
+             sep = ""), xlab = "log(mg kg-1)")
+     rug(log(jura.cal@data[, i]))
+     lines(density(log(jura.cal@data[, i])), col = "red")
+ }
> par(mfrow = c(1, 1))
```



Task 6 : Check the skewness of the transformed variables: •

```
> for (i in c(3:5, 9)) {
+     print(paste("log(", names(jura.cal@data[i]), ") : ",
+         round(skew(log(jura.cal@data[, i])), 3), sep = ""))
+ }
```

```
[1] "log(Cd) : -0.265"
[1] "log(Cu) : 0.323"
[1] "log(Pb) : 0.849"
[1] "log(Zn) : -0.189"
```

Q4 : *Are the log-transformed variables approximately normally-distributed? Which has the lowest skewness and most symmetric log-transformed histogram?* *Jump to A4*

•

We will pick one of these log-transformed variables to work on. From the histograms and skewness computation, it seem that log(Zn) is most suitable.

Task 7 : Add the transformed variable log(Zn) to the data frames. •

The assignment operator `<-` can create a new field in a data frame:

```
> jura.all@data$lZn <- log(jura.all@data$Zn)
> jura.cal@data$lZn <- log(jura.cal@data$Zn)
> jura.val@data$lZn <- log(jura.val@data$Zn)
> str(jura.cal@data)
```

```
'data.frame':        259 obs. of  10 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  1.74 1.33 1.61 2.15 1.56 ...
 $ Cu  : num  25.72 24.76 8.88 22.7 34.32 ...
 $ Pb  : num  77.4 77.9 30.8 56.4 66.4 ...
 $ Co  : num  9.32 10 10.6 11.92 16.32 ...
 $ Cr  : num  38.3 40.2 47 43.5 38.5 ...
 $ Ni  : num  21.3 29.7 21.4 29.7 26.2 ...
 $ Zn  : num  92.6 73.6 64.8 90 88.4 ...
 $ lZn : num  4.53 4.3 4.17 4.5 4.48 ...
```

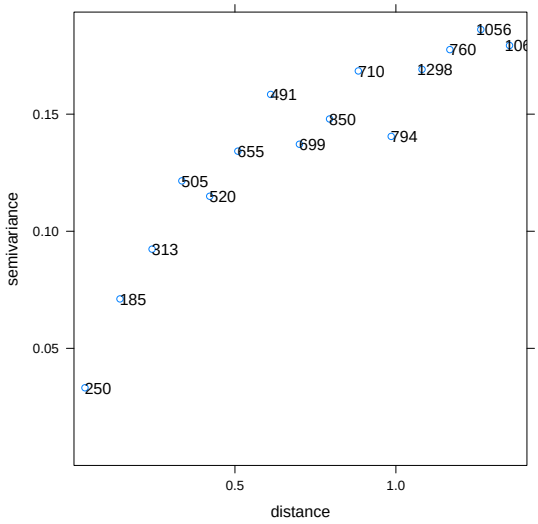
We proceed to model and interpolate one of the transformed variables, log(Zn), with OK as in Exercise 4.

2.3 Variogram modelling

Now we are able to compute and model the local spatial dependence for the transformed variable.

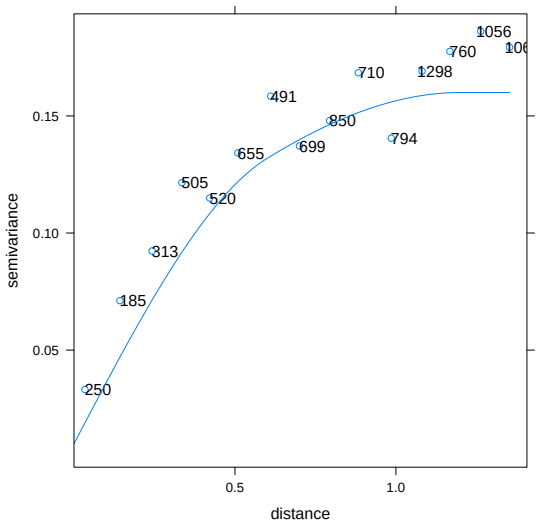
Task 8 : Compute the empirical variogram of log(Pb) in the calibration dataset and fit a variogram model. •

```
> v <- variogram(lZn ~ 1, jura.cal, cutoff = 1.4)
> print(plot(v, pl = T))
```

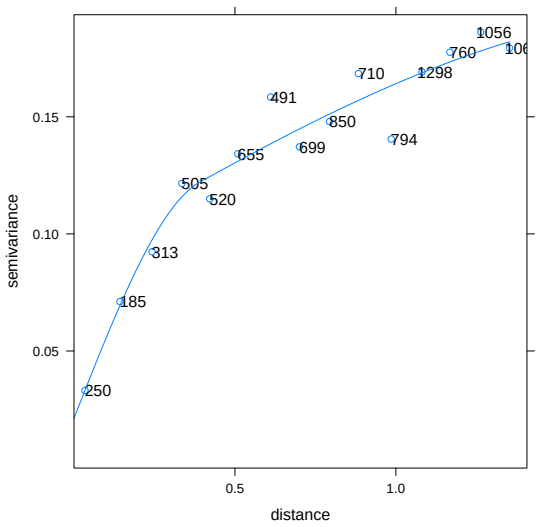


A double-spherical model is indicated from the strong "knick-point" near the second variogram bin (about 150 m separation); this is a sign of two spatial processes, one short- and one long-range.

```
> vm <- vgm(0.06, "Sph", 0.6, 0.01)
> vm <- vgm(0.09, "Sph", 1.2, add.to = vm)
> print(plot(v, pl = T, model = vm))
```




```
> (vmf <- fit.variogram(v, vm))

  model    psill   range
1   Nug 0.021300 0.00000
2   Sph 0.070067 0.39013
3   Sph 0.104176 1.96131

> print(plot(v, pl = T, model = vmf))
```



2.4 Ordinary Kriging

Now that the spatial dependence has been modelled, we can use it to predict at unsampled locations.

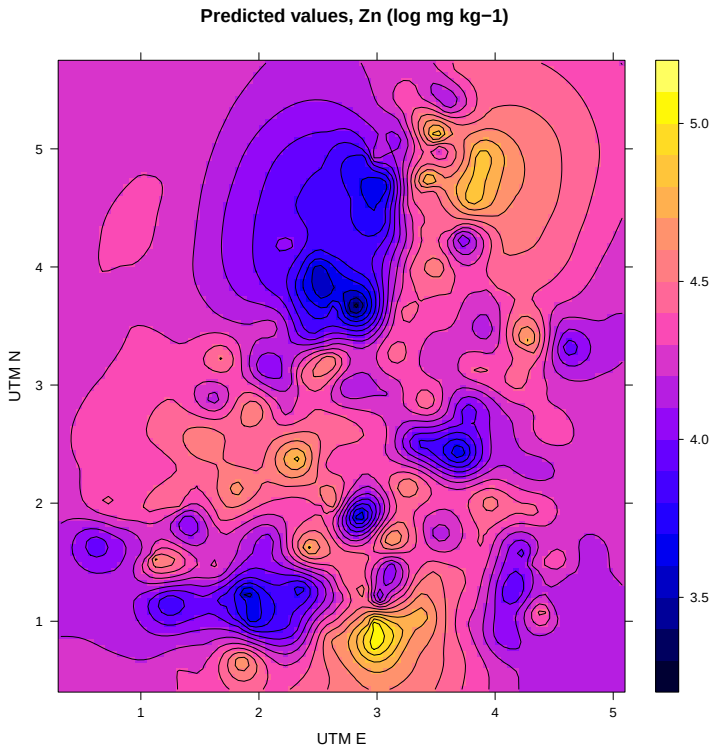
Task 9 : Predict log(Zn) on the prediction grid by OK. •

```
> k.lZn <- krige(lZn ~ 1, jura.cal, jura.grid, model = vmf)
```

```
[using ordinary kriging]

> summary(k.lZn)

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.   :3.37   Min.   :0.0299
 1st Qu.:4.16   1st Qu.:0.0858
 Median :4.28   Median :0.1097
 Mean   :4.27   Mean   :0.1247
 3rd Qu.:4.38   3rd Qu.:0.1671
 Max.   :5.06   Max.   :0.2074
```

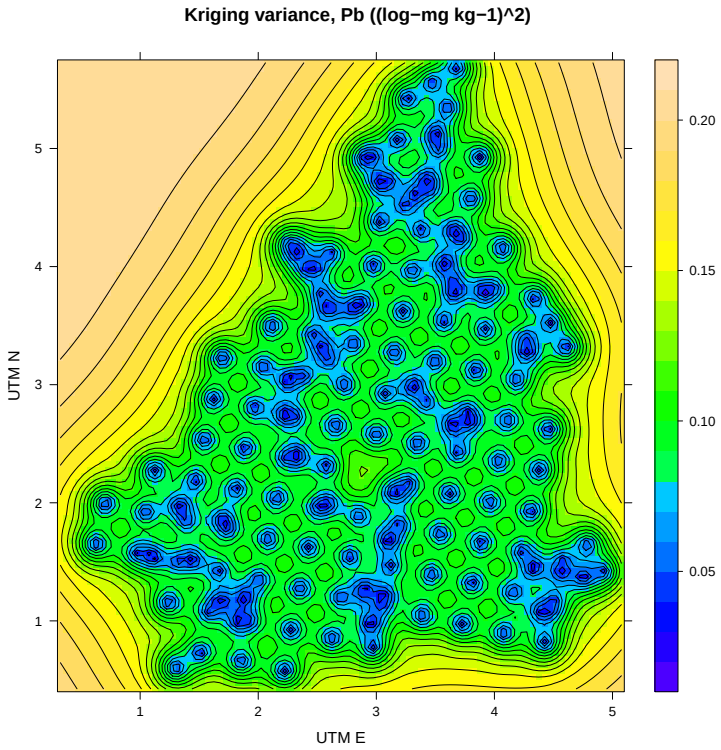
Task 10 : Display the map of the predictions: •

```
> print(spplot(k.lZn, zcol = "var1.pred", pretty = T, contour = T,
+     col.regions = bpy.colors(64), main = "Predicted values, Zn (log mg kg-1)",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T)))
```



Task 11 : Display the map of the prediction variances: •

```
> print(spplot(k.lZn, zcol = "var1.var", pretty = T, contour = T,
+     col.regions = topo.colors(64), main = "Kriging variance, Pb ((log-mg kg-1)^2)",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T)))
```



These are in log-units; i.e. $\log(\text{mg kg}^{-1})$ for the target variable and $(\log(\text{mg kg}^{-1}))^2$ for the kriging prediction variances.

3 Back-transformation

We now have kriged estimates of log(Zn). However, for many purposes we want the estimate of Zn itself.

> **Note:** For example, regulatory thresholds are expressed in concentrations of the metal. Any theshold value can be log-transformed itself and then used to classify the result of lognormal kriging into above/below the threshold. However this does not result in a continuous map of the untransformed variable.

Thus we often must **back-transfom** the kriged estimate. Naïvely we might just reverse the transformation:

$$\text{Zn} = e^{\log \text{Zn}} \tag{6}$$

! → But this is **not correct**. The estimate log(Zn) is a **weighted sum** of sample values of log(Zn); and of course the sum of logarithms is in fact a product. So the whole weighting scheme was different than if we had used the original values.

The correct back-transformation was derived by Journel [3]; the formulas are also presented by Webster and Oliver [4, §8.10] and Goovaerts [2], but with no derivation.

In the case of Simple Kriging (where the mean is known), it is possible to back-transform both the estimates and their variances; but for OK, where the true spatial mean is not known, only the predicted value can be derived.

To derive the predicted values of target variable Z from its log-transform $Y = \log(Z)$ at prediction location $\mathbf{x}_0$:

$$\hat{Z}_{\mathrm{OK}}(\mathbf{x}_0) \quad = \quad \exp\left[\hat{Y}_{\mathrm{OK}}(\mathbf{x}_0) + \sigma^2_{\mathrm{OK}}(\mathbf{x}_0)/2 - \psi_{\mathrm{OK}}\right] \tag{7}$$

where ψ_{OK} is the LaGrange multiplier in the solution of the OK system.

However, `gstat` does not use minimization with LaGrange multipliers to solve the kriging system, so this formula can not be applied. Instead, `gstat` uses a *regression* approach to kriging:

1. `gstat` first solves for the *spatial mean* (or, for UK, the regional trend) using *Generalized Least Squares* (GLS), which requires knowledge of the covariance structure revealed by variogram modelling to account for spatial correlation of the observations;
2. `gstat` then applies Simple Kriging (SK) on the *residuals* from this mean (or trend);
3. The final prediction at a point is the sum of the spatial mean (or trend) and the predicted residual at that point.

See Exercise 4, §6 for details of the computation, which are revealed by use of the optional `debug.level` argument to the `krige` method.

Both the computation of the spatial mean (or trend) and SK give a standard error of prediction; these can be added for a final standard error.

For Simple Kriging (SK), the back-transformation does not include the LaGrange multiplier:

$$\hat{Z}_{\mathrm{SK}}(\mathbf{x}_0) \quad = \quad \exp\left[\hat{Y}_{\mathrm{SK}}(\mathbf{x}_0) + \sigma^2_{\mathrm{SK}}(\mathbf{x}_0)/2\right] \tag{8}$$

In the case of SK the kriging prediction variance can also be back-transformed:

$$\mathrm{var}_{\mathrm{SK}}(\mathbf{x}_0) \quad = \quad \mu_Z^2 \exp(\sigma^2_{\mathrm{SK}})\left[1 - \exp\{-\sigma^2_{\mathrm{SK}}(\mathbf{x}_0)/2\}\right] \tag{9}$$

where μ_Z is the expected value of $Z(\mathbf{x}_0)$. In SK we must know this; here we don't know it; instead we are estimating as the spatial mean. This in turn is back-transformed from the BLUE prediction of the log-transformed spatial mean μ_Y as:

$$\mu_Z \quad = \quad \exp\left[\mu_Y + \sigma^2(\mu_Y)/2\right] \tag{10}$$

where μ_Y is the BLUE prediction of the spatial mean, and $\sigma^2(\mu_Y)$ is its prediction variance; this is the same formula for back-transformation of a log-transformed variable used above for the SK predictions.

The BLUE prediction is returned if the optional `BLUE` argument to the `krige` method is set to `TRUE`, so that only the best linear unbiased estimator (BLUE) of the parameter is returned; by default this argument is `FALSE`, so that the best linear unbiased predictor (BLUP) of the point is returned. Normally the BLUP is what we want, i.e. the predicted value at the point. The BLUE gives us the spatial mean, i.e. the expected value withough taking any nearby points into special account.

> **Note:** In the case of Universal Kriging (UK), the BLUE at any point would be the trend-surface coefficients.

Task 12 : Compute the BLUE for the transformed target variable. •

Unfortunately, the `krige` method does not allow us to request only the BLUE; this is a more sophisticated option of the `gstat` package. So, the `gstat` method (of the `gstat` package!) must be used.

First we have to create a new `gstat` object with the `gstat` method; we indicate it's a new object by specifying a `NULL` first argument:

! → **Warning!** The use of `gstat` is not so clear, pay close attention.

```
> (g <- gstat(NULL, id = "lZn", form = lZn ~ 1, data = jura.cal,
+     model = vmf))

data:
lZn : formula = lZn~~1 ; data dim = 259 x 10
variograms:
       model    psill   range
lZn[1]   Nug 0.021300 0.00000
lZn[2]   Sph 0.070067 0.39013
lZn[3]   Sph 0.104176 1.96131
```

Notice this object has all the information so far about this variable: the data, the locations, and the variogram model.

Now we can use the `predict.gstat` method. It is sufficient to predict at one point, since all have the same BLUE for OK (no trend). So we'll predict for the first point in the grid. We extract this with the usual `[` operator, but must keep the whole grid topology by specifying the optional `drop=FALSE` argument. Then we convert to a single spatial point with the `SpatialPoints` method; this can then be used as new data for prediction.

```
> jura.one <- SpatialPoints(jura.grid[1, drop = FALSE])
> k.lZn.blue <- predict.gstat(g, newdata = jura.one, BLUE = T)

[generalized least squares trend estimation]

> k.lZn.blue$lZn.pred

[1] 4.273
```

```
> k.lZn.blue$lZn.var

[1] 0.011848
```

Q5 : *What is the expected value and its standard error?* *Jump to A5* •

Task 13 : Back-transform the expected value to original units. •

```
> (Zn.blue <- exp(k.lZn.blue$lZn.pred + k.lZn.blue$lZn.var/2))

[1] 72.166
```

Q6 : *How does this compare with the non-spatial mean?* *Jump to A6* •

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

[1] 75.078
```

Task 14 : Compute the SK prediction and its variance for the residuals. •

We first compute the residuals at the calibration points and add them to that data frame:

```
> jura.cal$lZn.res <- jura.cal$lZn - k.lZn.blue$lZn.pred
> summary(jura.cal$lZn.res)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.0500 -0.2660  0.0251 -0.0272  0.2260  1.1200
```

Q7 : *Why is the mean residual not zero?* *Jump to A7* •

To predict, we need a variogram for the residiuals.

Q8 : *Is it necessary to re-compute and re-model the variogram?* *Jump to A8* •

Finally, we use SK to predict, with the known (spatial) mean of zero; SK will be used if the optional `beta` argument to the `krige` method is specified:

```
> k.lZn.res.sk <- krige(lZn.res ~ 1, jura.cal, newdata = jura.grid,
+     model = vmf, beta = 0)

[using simple kriging]

> summary(k.lZn.res.sk)

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.   :-0.90203   Min.   :0.0299
 1st Qu.:-0.11050   1st Qu.:0.0858
 Median : 0.00440   Median :0.1094
 Mean   :-0.00731   Mean   :0.1225
 3rd Qu.: 0.11065   3rd Qu.:0.1641
 Max.   : 0.78716   Max.   :0.1955
```

This kriging object contains the prediction as field `var1.pred` and its variance as field `var1.var`. Note that the variance is close to, but not identical to, that for the original values:

```
> summary(round(k.lZn.res.sk$var1.var - k.lZn$var1.var,
+     3))

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-0.01200 -0.00300  0.00000 -0.00223  0.00000  0.00000
```

This discrepency is likely caused by the difference between the spatial and non-spatial means.

We now have all the information to compute the back-transformation.

Task 15 : Back-transform the residuals and their prediction variances to original units. •

To back-transform the residuals, we use Equation 8:

```
> k.lZn.res.sk$res.pred <- exp(k.lZn.res.sk$var1.pred +
+     k.lZn.res.sk$var1.var/2)
> summary(k.lZn.res.sk$res.pred)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.412   0.954   1.090   1.080   1.180   2.240
```

To back-transform the variance, we use Equation 10, which includes the back-transformed expected value.

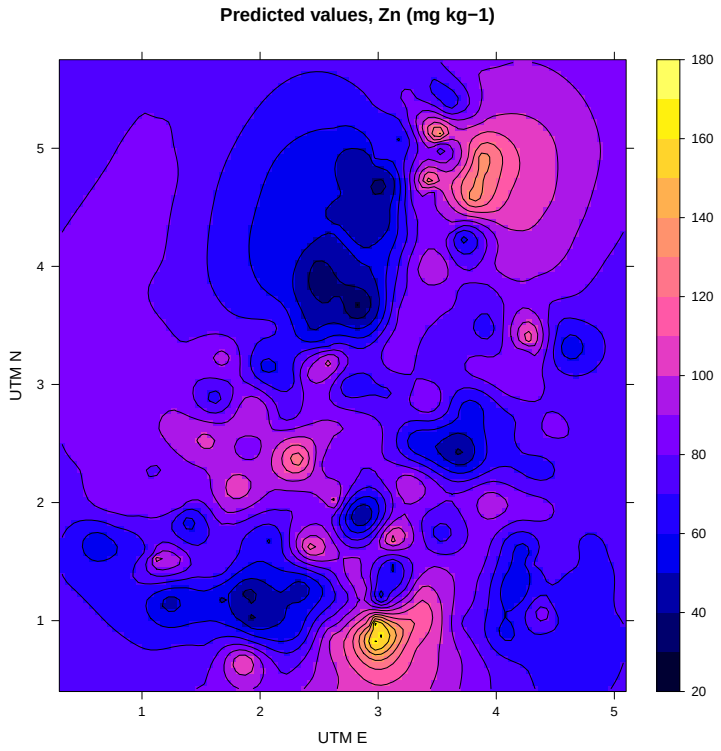
```
> (mu.Z <- exp(k.lZn.blue$lZn.var/2))

[1] 1.0059

> k.lZn.res.sk$res.var <- mu.Z^2 * exp(k.lZn.res.sk$var1.var) *
+     (1 - exp(-k.lZn.res.sk$var1.var)/2)
> summary(k.lZn.res.sk$res.var)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.537   0.597   0.623   0.639   0.686   0.725
```

Finally, we can multiply the back-transformed spatial mean by the back-transformed residuals to get the BLUP in original units. Note that the variance is the same for the residuals or for the original values.

```
> print(Zn.blue)

[1] 72.166

> k.lZn$Zn.pred <- Zn.blue * k.lZn.res.sk$res.pred
> summary(k.lZn$Zn.pred)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   29.7    68.9    78.6    77.7    85.1   162.0

> k.lZn$Zn.var <- k.lZn.res.sk$res.var
> summary(k.lZn$Zn.var)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.537   0.597   0.623   0.639   0.686   0.725
```

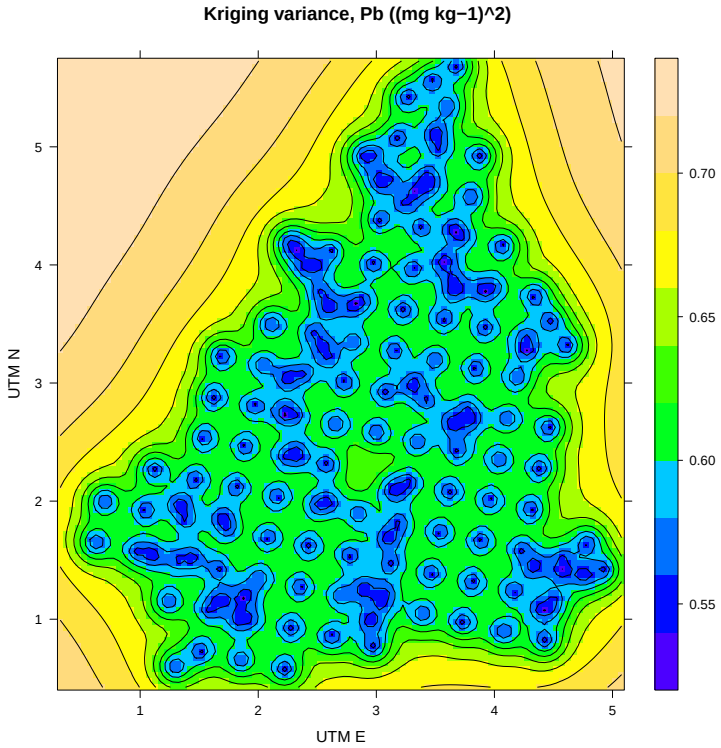
Task 16 : Display the predictions and variances. •

```
> print(spplot(k.lZn, zcol = "Zn.pred", pretty = T, contour = T,
+     col.regions = bpy.colors(64), main = "Predicted values, Zn (mg kg-1)",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T)))
```



Task 17 : Display the map of the prediction variances: •

```
> print(spplot(k.lZn, zcol = "Zn.var", pretty = T, contour = T,
+     col.regions = topo.colors(64), main = "Kriging variance, Pb ((mg kg-1)^2)",
+     xlab = "UTM E", ylab = "UTM N", scales = list(draw = T)))
```



4 Clean up

Task 18 : Remove the temporary variables created in this exercise. •

```
> rm(skew, g, i, v, vm, vmf, k.lZn, k.lZn.blue, k.lZn.res,
+     k.lZn.res.sk, Zn.blue, mu.Z)
```

5 Answers

A1 : *Fields 3 through 9:* `Cd`, `Cu`, `Pb`,`Co`, `Cr`, `Ni`, `Zn`. *Return to Q1* •

A2 : `Cd`, `Cu`, `Pb`, and `Zn`. *Return to Q2* •

A3 : *As before,* `Cd`, `Cu`, `Pb`, and `Zn`. *Return to Q3* •

A4 : *Yes, they are all much closer to non-skewed (symmetric). Log(Zn) has the lowest absolute skewness and looks most symmetric in the histogram plot.* *Return to Q4* •

A5 : 4.273 ± 0.1088 *Return to Q5* •

A6 : *The non-spatial mean is 75.08; the spatial mean is somewhat smaller.* *Return to Q6* •

A7 : *Because the mean we subtracted from the original values is the spatial mean (back-transformed), not the non-spatial mean.* *Return to Q7* •

A8 : *No, because we subtracted the same value (the spatial mean) from each; this will not change the semi-variance of point pairs, so the residual variogram will be the same.* *Return to Q8* •

References

[1] M G Bulmer. *Principles of statistics*. Dover Publications, New York, 1979. 4

[2] P Goovaerts. *Geostatistics for natural resources evaluation*. Applied Geostatistics. Oxford University Press, New York; Oxford, 1997. 1, 11

[3] A G Journel. The lognormal approach to predicting local distributions of selective mining unit grades. *Mathematical Geology*, 12(4):285–303, 1980. 1, 11

[4] R Webster and M A Oliver. *Geostatistics for environmental scientists*. Wiley & Sons, Chichester, 2001. 1, 11

Index of R Concepts

`<-` operator, 6
`[` operator, 12

`beta` method argument, 13
`BLUE` method argument, 12

`debug.level` method argument, 11

`e1071` package, 4

`gstat (package:gstat)`, 12
`gstat` package, 1, 11, 12

`krige (package:gstat)`, 11–13
`kurtosis`, 4

`load`, 1, 2

`predict.gstat (package:gstat)`, 12

`setwd`, 2
`skew`, 4
`sp` package, 1
`SpatialPoints (package:sp)`, 12