Spatial Statistics

Computer lab – Session 4

Madlene Nussbaum Andreas Papritz

21 Oct 2024

Table of contents

1 Geostatistical analysis of lognormal data with anisotropy

1.1 Exploratory analysis

Copper content in soils around metal smelter in Dornach

Data:

- dornach.txt
- dornach_grid.txt

Read the data file dornach.txt into R. The data set contains the following variables:

- `x`, `y`: anonymized coordinates of sites where soil samples were taken, the origin of the coordinate system is the main stack of the smelter,
- `survey`: factor coding the survey in which the data was collected,

- `forest`: factor coding whether the site is in a forest,
- `built.up`: factor coding when a site has been built up,
- `geology`: factor coding the geologic parent material at a site,
- `cu`, `cd`, `zn`: heavy metal concentration (mg/kg) in topsoil (0–20 cm depth).

We will predict topsoil copper content `cu` to evaluate the pollution situation over the study are.

> Task 1
>
> Plot the spatial distribution of the sites with observed `cu` and its value, use e.g. a bubble plot.

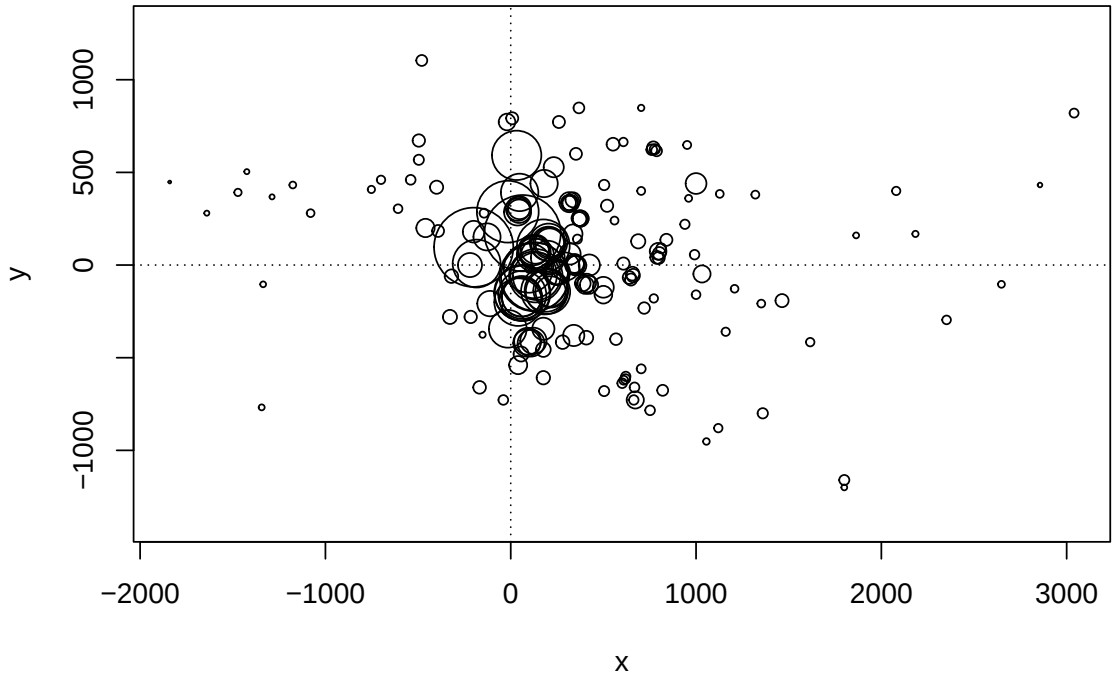
> Task 2
>
> Add the distance to the smelter as a new column to the data. Explore the distribution of `cu`. Find a suitable transformation for both `cu` and the distance to approximately linearize the relationship.

> Task 3
>
> Explore the directional relationship. Compute the angle to the smelter by `atan2(x,y)` and convert the angle to a factor using `cut()`, e.g. differentiate North/East/South/West. Plot the angles with different colors per directional group.

Solution Task 1

```
d.dornach <- read.table("data/dornach.txt", header=TRUE, stringsAsFactors = TRUE)
```

```
plot(y~x, d.dornach, cex=sqrt(cu)/10, asp=1)
abline(h=0, lty="dotted"); abline(v=0, lty="dotted")
```



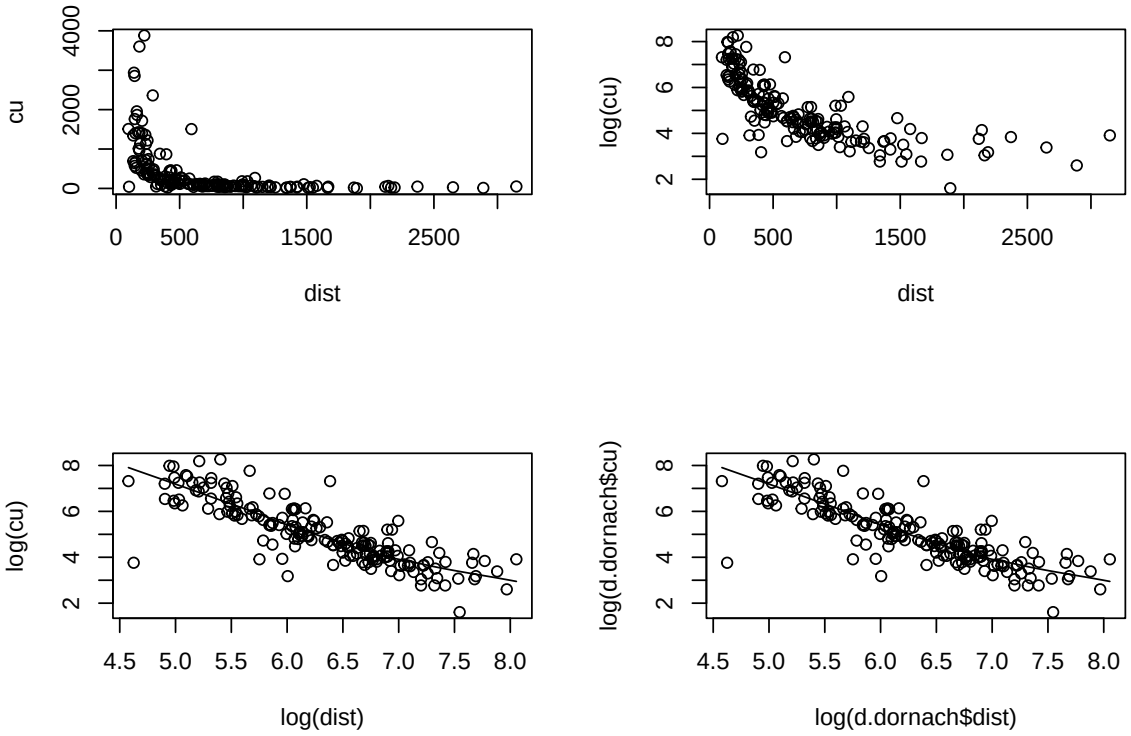
There are very larger values nearby the smelter. Further away, less samples are available, but the cu concentration also drops.

Solution Task 2

Compute distance to smelter:

```
d.dornach$dist <- with(d.dornach, sqrt(x^2 + y^2))
```

```
par(mfrow=c(2, 2))
plot(cu~dist, d.dornach)
plot(log(cu)~dist, d.dornach)
with(d.dornach, scatter.smooth(log(cu)~log(dist)))
scatter.smooth(log(d.dornach$cu)~log(d.dornach$dist))
```



Dependence of `cu` on `dist`

`cu` concentration is largest close to the main stack of the smelter which is at x = 0 and y = 0. The plot of `cu` vs. `dist` shows a sharp decrease of `cu` with increasing dist. Furthermore, for short distances there is much more variation in `cu` than for long distances. We mend this issue by plotting `log(cu)` against `dist`. The plot shows now an (approximately) exponential decay of `log(cu)` with `dist` and we linearize the relation by log-transforming `dist` as well.

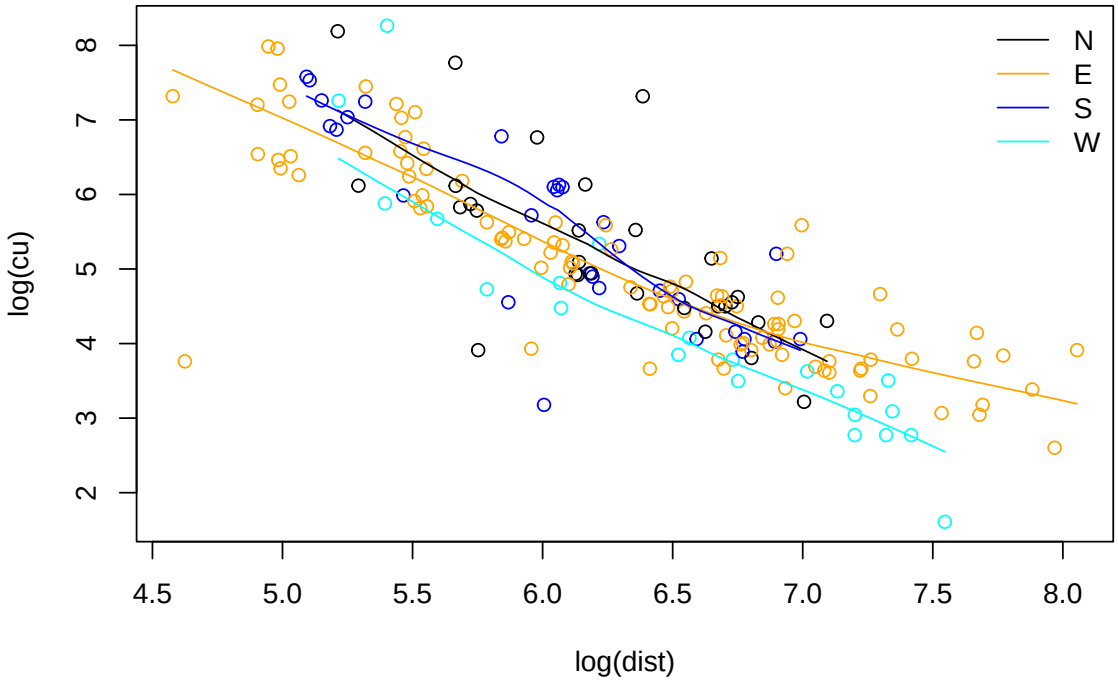
Solution Task 3

Compute orientation to main stack of smelter (coordinates 0,0):

```
d.dornach$angle <- with(d.dornach, atan2(y,x))
d.dornach$sector.4 <- as.numeric(cut(d.dornach$angle,
  breaks=pi*c(-1, -0.75, -0.25, 0.25, 0.75, 1)))
d.dornach$sector.4[d.dornach$sector.4==5] <- 1
d.dornach$sector.4 <- factor(d.dornach$sector.4, levels = 4:1,
                             labels = c("N", "E", "S", "W"))
```

Next, we color the points in the plot of `log(cu)` vs. `log(dist)` by sector.4 and smooth the points sector-wise by `loess()`:

```
palette(c("black", "orange", "blue", "cyan"))
plot(log(cu)~log(dist), d.dornach, col=sector.4)
for( i in 1:nlevels(d.dornach$sector.4)){
  sel <- with(d.dornach, sector.4 == levels(sector.4)[i])
  lines(with(d.dornach, loess.smooth(log(dist)[sel], log(cu)[sel])), col=i)
}
legend("topright", lty=1, col=1:nlevels(d.dornach$sector.4),
  legend=levels(d.dornach$sector.4), bty="n")
```



The `loess()` smooths for the 4 sectors run approximately parallel but are vertically shifted: For a given distance concentrations are largest for sectors N & S followed by E and W. Hence, for a given distance, concentrations between sectors N & S and W differ on average by a factor of about exp(1) 3. This suggests that we should take orientation into account when modelling the trend. As the `loess()` lines are approximately parallel we include only a direction-dependent intercept in the regression model (but not a direction-dependent slope).

One possibility would be to include the factor sector.4 in the model. However, this would result in a non- continuous trend surface with discontinuities at the sector boundaries.

1.2 Fit a trend model

> Task 1
>
> Fit a ordinary least squares regression using distance and angle as explored above. Use `cos()` and `sin()` to transform the angle in continuous covariates (results in a periodic function).

> Task 2
>
> Extract the residuals from this regression model and check whether `cu` further depends on the land use (variables `forest`, `built.up`), on the parent material (`geology`) and on the origin of the data (`survey`)? Use for example boxplots with notches. If necessary update the regression model by adding those covariates that seem to influence `cu`.

> Task 3
>
> Assess the fit of the regression model by the usual residual diagnostics plots and display the spatial distribution of the residuals by a bubble plot.

Solution Task 1

```
r.lm.0 <- lm(log(cu) ~ log(dist) + cos(angle) + sin(angle), d.dornach)
summary(r.lm.0)
```

```
Call:
lm(formula = log(cu) ~ log(dist) + cos(angle) + sin(angle), data = d.dornach)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8196 -0.3654 -0.1141  0.3179  2.5253

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 14.17023    0.44372  31.935   <2e-16 ***
log(dist)   -1.46553    0.06963 -21.047   <2e-16 ***
```

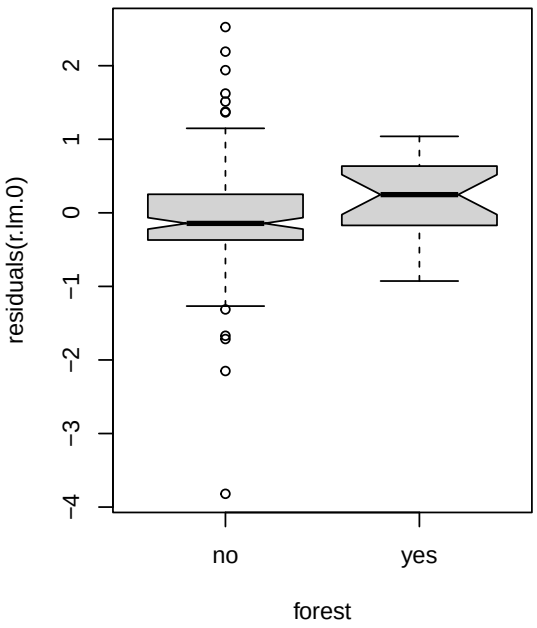
```
cos(angle)   0.18586    0.08278   2.245    0.026 *
sin(angle)  -0.03102    0.08803  -0.352    0.725
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7084 on 177 degrees of freedom
Multiple R-squared:  0.7194,    Adjusted R-squared:  0.7146
F-statistic: 151.2 on 3 and 177 DF,  p-value: < 2.2e-16
```

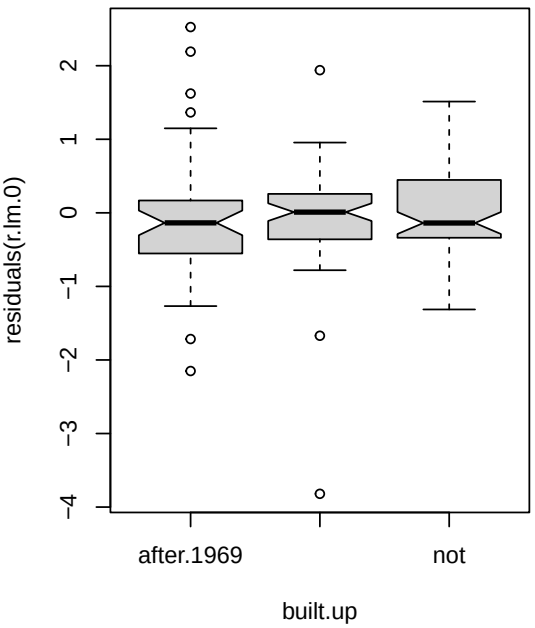
Solution Task 2

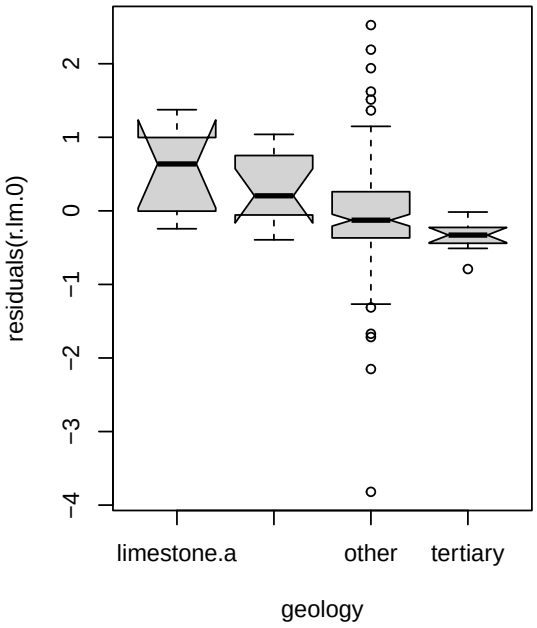
```
par(mfrow=c(2,2))
plot(residuals(r.lm.0)~forest, d.dornach, notch=TRUE)
plot(residuals(r.lm.0)~built.up, d.dornach, notch=TRUE)
plot(residuals(r.lm.0)~geology, d.dornach, notch=TRUE)
```

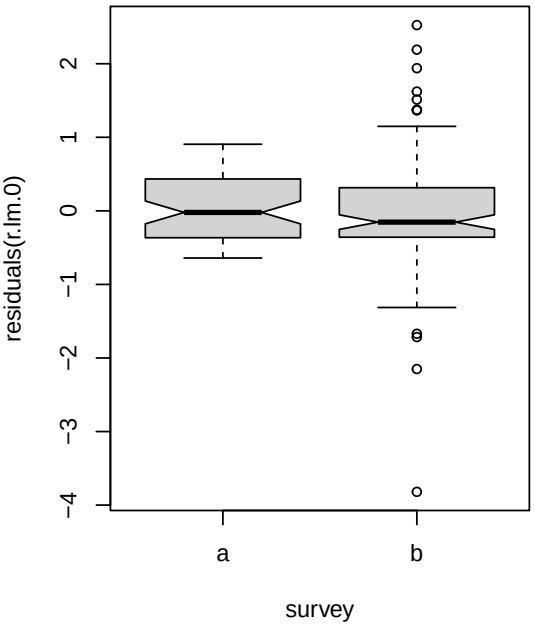
```
Warning in (function (z, notch = FALSE, width = NULL, varwidth = FALSE, : some
notches went outside hinges ('box'): maybe set notch=FALSE
```

```
plot(residuals(r.lm.0)~survey, d.dornach, notch=TRUE)
```


residuals(r.lm.0)
2
1
0
−1
−2
−3
−4
no
yes
forest


residuals(r.lm.0)
2
1
0
−1
−2
−3
−4
after.1969
not
built.up


residuals(r.lm.0)
2
1
0
−1
−2
−3
−4
limestone.a
other
tertiary
geology


residuals(r.lm.0)
2
1
0
−1
−2
−3
−4
a
b
survey

There might be a weak dependence on forest and geology. We add therefore these covariates to the model:

```
r.lm.1 <- update(r.lm.0, .~. + forest + geology)
summary(r.lm.1)
```

```
Call:
lm(formula = log(cu) ~ log(dist) + cos(angle) + sin(angle) +
    forest + geology, data = d.dornach)

Residuals:
    Min      1Q  Median      3Q     Max
-3.9381 -0.3294 -0.0428  0.3138  2.5341

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        15.661489   0.716400  21.861   <2e-16 ***
log(dist)          -1.592095   0.086969 -18.306   <2e-16 ***
cos(angle)          0.130918   0.087229   1.501   0.1352
sin(angle)          0.006255   0.092776   0.067   0.9463
forestyes           0.051754   0.234277   0.221   0.8254
geologylimestone.b -0.199570   0.339470  -0.588   0.5574
geologyother       -0.726249   0.322084  -2.255   0.0254 *
geologytertiary    -0.923083   0.356525  -2.589   0.0104 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6925 on 173 degrees of freedom
Multiple R-squared:  0.7379,    Adjusted R-squared:  0.7273
F-statistic: 69.56 on 7 and 173 DF,  p-value: < 2.2e-16
```

Whereas forest is not significant, some levels of geology seem to differ and we keep this variable for the time being in the model, although the goodness-of-fit improved only marginally.

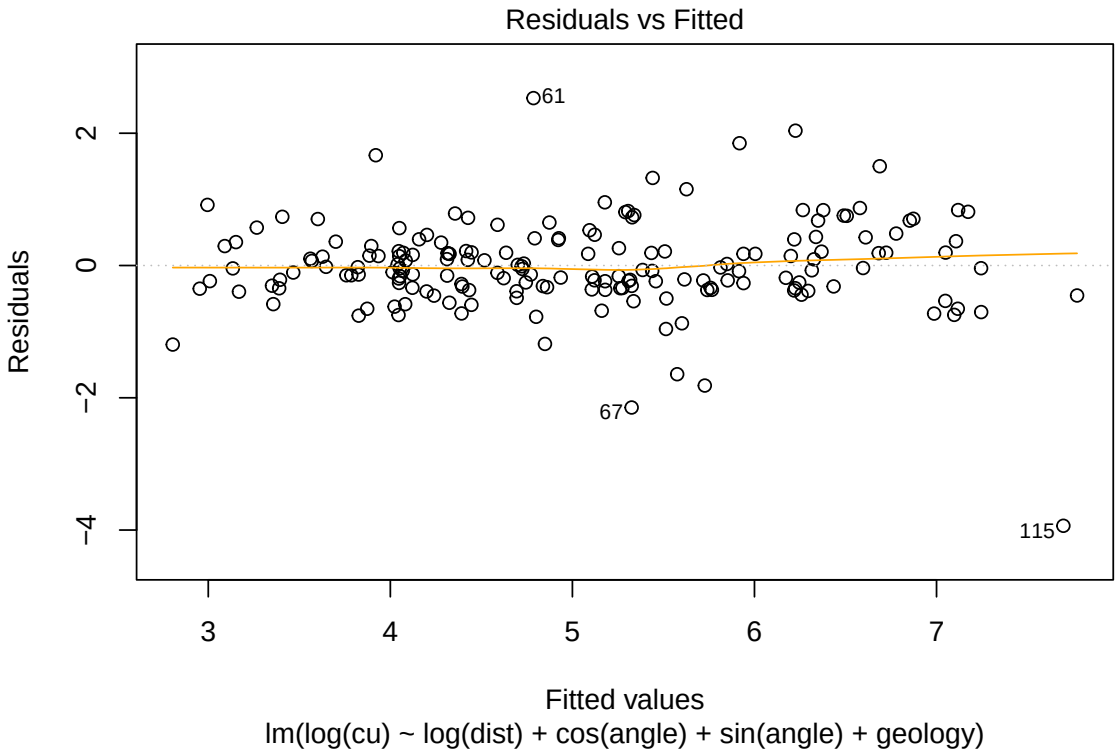
```
r.lm.2 <- update(r.lm.1, .~. - forest)
summary(r.lm.2)
```

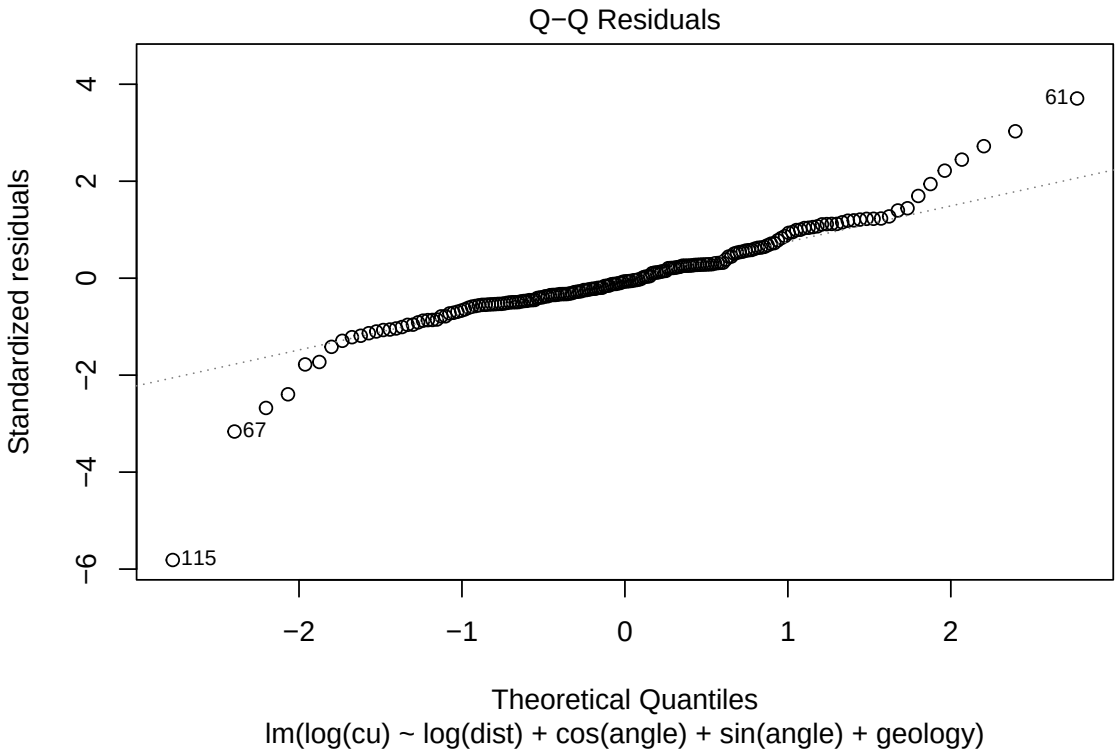
```
Call:
lm(formula = log(cu) ~ log(dist) + cos(angle) + sin(angle) +
    geology, data = d.dornach)
```

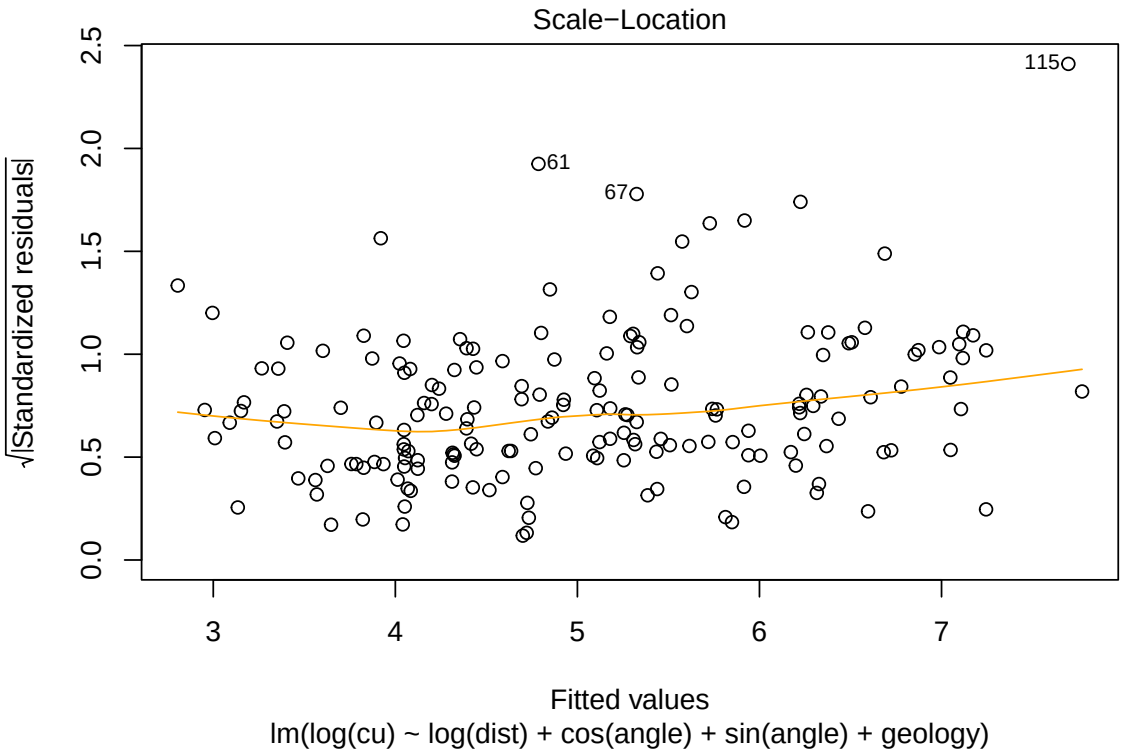
```
Residuals:
    Min      1Q  Median      3Q     Max
-3.9356 -0.3386 -0.0444  0.3459  2.5310

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        15.682236   0.708273  22.142  < 2e-16 ***
log(dist)          -1.590441   0.086409 -18.406  < 2e-16 ***
cos(angle)          0.126856   0.085036   1.492  0.13757
sin(angle)          0.005361   0.092434   0.058  0.95382
geologylimestone.b -0.181828   0.328930  -0.553  0.58112
geologyother       -0.753350   0.296990  -2.537  0.01207 *
geologytertiary    -0.947709   0.337722  -2.806  0.00558 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6906 on 174 degrees of freedom
Multiple R-squared:  0.7378,    Adjusted R-squared:  0.7287
F-statistic:  81.6 on 6 and 174 DF,  p-value: < 2.2e-16
```

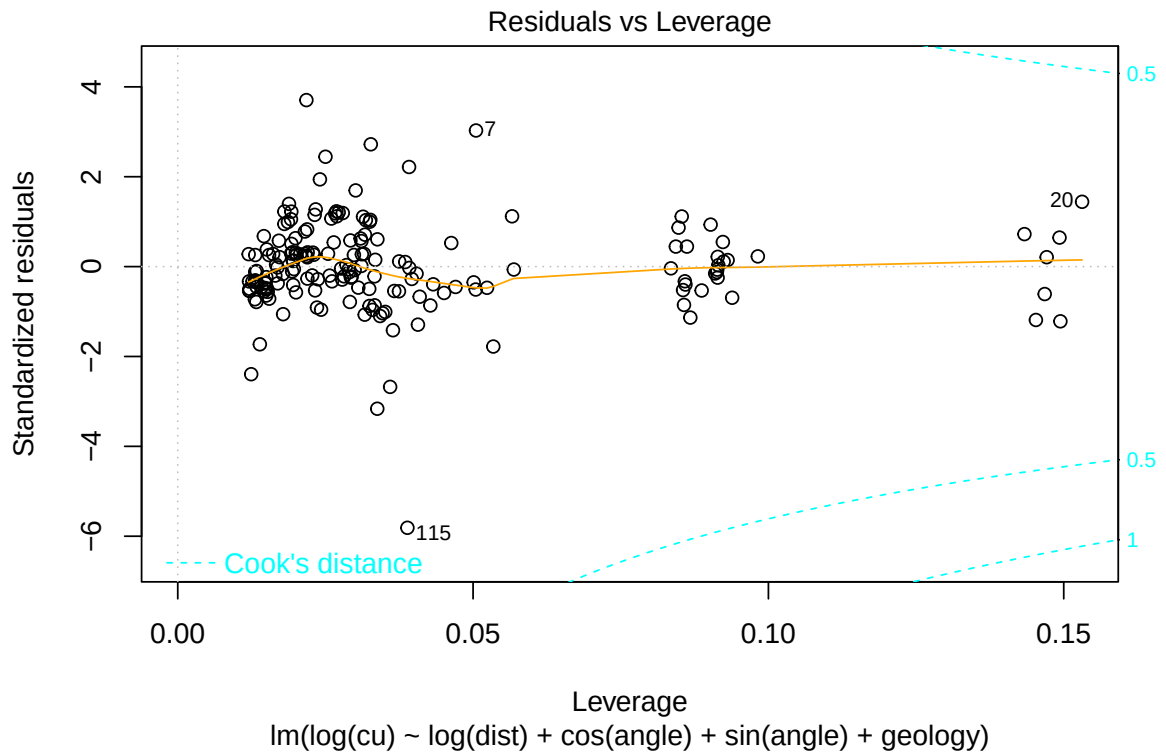
Solution Task 3

```r
plot(r.lm.2)
```

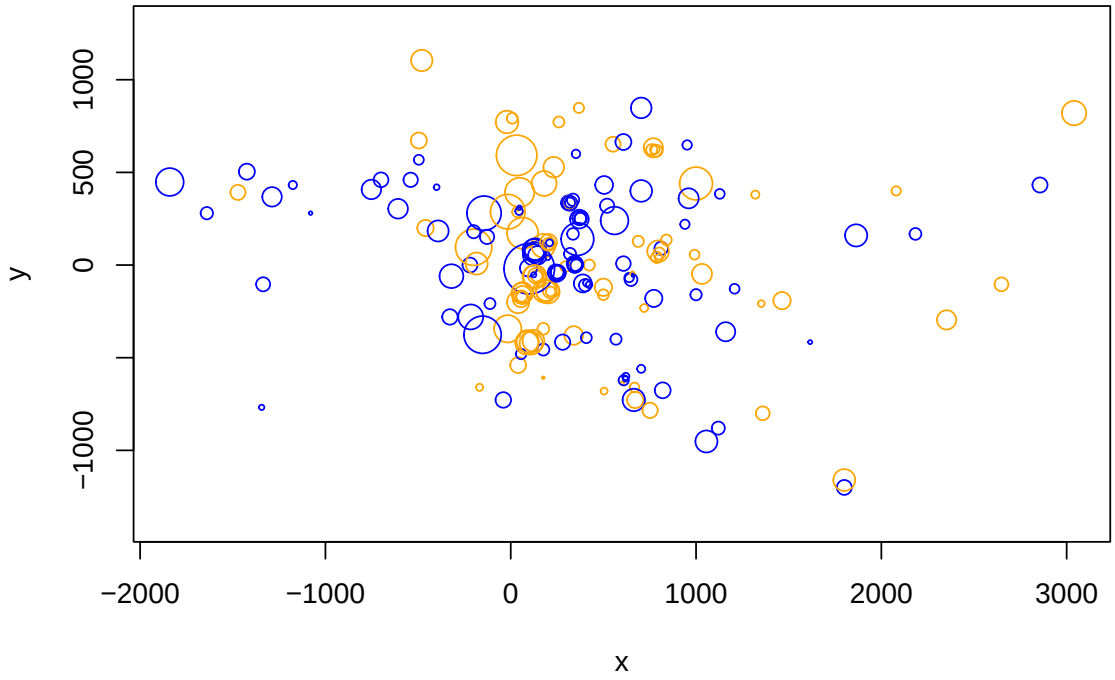

Residuals vs Fitted
Residuals
Fitted values
lm(log(cu) ~ log(dist) + cos(angle) + sin(angle) + geology)
61
67
115
2
0
−2
−4
3
4
5
6
7


Q−Q Residuals
Standardized residuals
4
2
0
−2
−4
−6
61
67
115
−2
−1
0
1
2
Theoretical Quantiles
lm(log(cu) ~ log(dist) + cos(angle) + sin(angle) + geology)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
115
61
67
3
4
5
6
7
Fitted values
lm(log(cu) ~ log(dist) + cos(angle) + sin(angle) + geology)




```
plot(y~x, d.dornach, asp=1, cex=2*sqrt(abs(residuals(r.lm.2))),
     col = c("blue", NA, "orange")[sign(residuals(r.lm.2))+2])
```



Apart from presence of some (mostly negative) outliers and auto-correlation of residuals at adjacent locations there are no obvious violations of the modelling assumptions apparent.

1.3 Explore variogram and fit REML

Task 1

Compute the sample variogram of the residuals of the regression model and fit an exponential variogram model to the sample variogram.

Task 2

Estimate the coefficients of the trend model fitted above and the parameters of the exponential variogram model now by Restricted Maximum Likelihood (REML) using the function `georob()`.

Task 3

Fit an anisotropic variogram model with `georob()`. Fit `f1` and `omega` and keep the other parameters fixed. Check example on the help page.

Hint

Try to find reasonable starting values: According to above directional plot, `cu` values in Western direction seem to be smaller, i.e. the decrease is larger than in the other directions. Therefore, it is likely `f1` $<$ `f2`. For `omega` we would need to do more plotting or try to fit with `georob()` and check if the numerical solving of the model converged. If not, we would need to find better starting values.

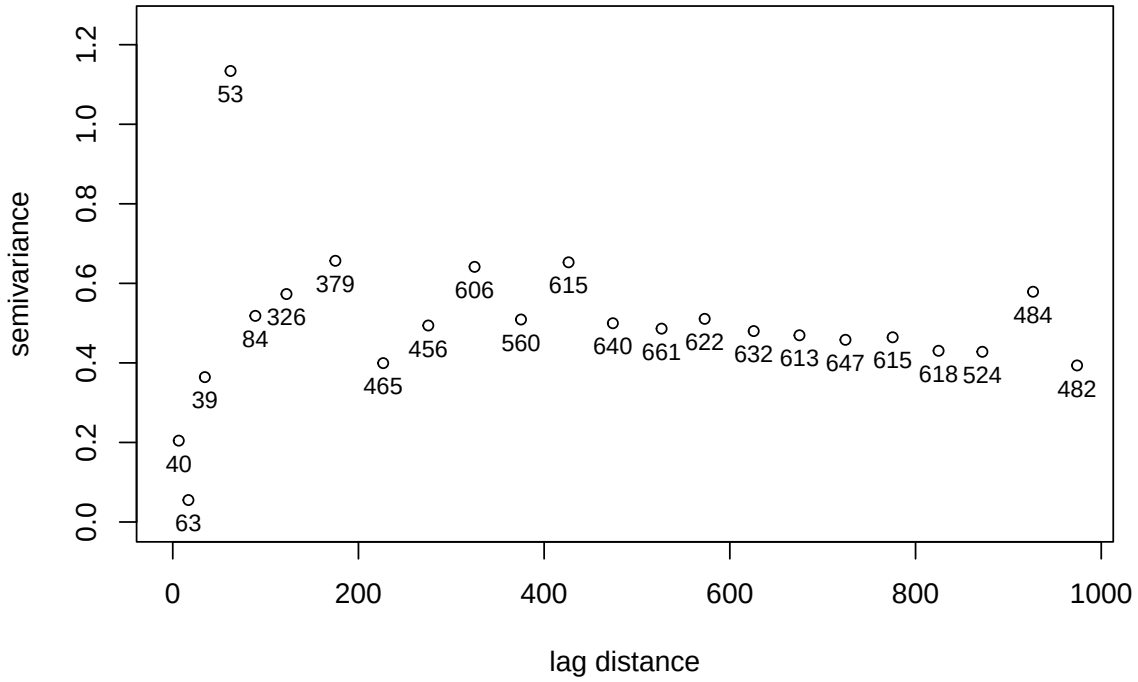
Task 4

Assess the fit of the model by residual diagnostic plots.

Note: we could now to further model selection and trying to improve the model also by adding interactions of the covariates. We could do some plotting and use `georob::step()`.

Solution Task 1

```
library(georob)
r.sv <- sample.variogram(residuals(r.lm.2),
                         locations=as.matrix(d.dornach[, c("x", "y")]),
                         estimator="matheron",
                         lag.dist.def=c(0, 10, 25, 50, 75, seq(100, 1000, by=50)))
plot(r.sv)
text(gamma~lag.dist, r.sv, labels=npairs, pos=1, cex=0.8)
```




```
r.sv.exp <- fit.variogram.model(r.sv, variogram.model="RMexp",
                                param=c(variance=0.4, nugget=0.05, scale=30))
r.sv.exp
```

```
Variogram:  RMexp
        variance    snugget(fixed)            nugget             scale
        0.520286          0.000000          0.002627         12.950799
```

From the sample variogram one estimates a range of about 100 m. However, the fitted range is much shorter, likely because of the large estimated semivariance for the 4th lag class. The estimated nugget is close to zero, the sill equals 0.45. Note that the range parameter estimated for an exponential variogram model is equal to about one third of the effective range read from a sample variogram. Except for the variogram models with compact support (e.g. spherical model family, cubic variogram model) the fitted range parameter is not equal to the effective range of the sample variogram, but of course is linearly related to it.

Solution Task 2

```
r.georob <- georob(log(cu) ~ log(dist) + cos(angle) + sin(angle) + geology,
                   data=d.dornach, locations=~x+y, variogram.model="RMexp",
                   param=c(variance=0.4, nugget=0.05, scale=30), tuning.psi=1000)
summary(r.georob)
```

```
Call:georob(formula = log(cu) ~ log(dist) + cos(angle) + sin(angle) +
    geology, data = d.dornach, locations = ~x + y, variogram.model = "RMexp",
    param = c(variance = 0.4, nugget = 0.05, scale = 30), tuning.psi = 1000)

Tuning constant:  1000

Convergence in 8 function and 8 Jacobian/gradient evaluations

Estimating equations (gradient)

                                  eta         scale
  Gradient           :  3.020576e-04 -6.196804e-03

Maximized restricted log-likelihood: -183.649

Predicted latent variable (B):
     Min       1Q   Median       3Q      Max
-1.92522 -0.26496 -0.02113  0.24321  1.63467

Residuals (epsilon):
     Min       1Q   Median       3Q      Max
-2.06663 -0.10711 -0.01433  0.11256  0.91312

Standardized residuals:
    Min      1Q  Median      3Q     Max
-6.7990 -0.3995 -0.0616  0.4807  3.3698


Gaussian REML estimates

Variogram:  RMexp
                Estimate    Lower   Upper
variance         0.33807  0.21373   0.535
snugget(fixed)   0.00000       NA      NA
nugget           0.16776  0.09349   0.301
```

```
scale          58.28482 25.42591 133.609


Fixed effects coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         16.0660     0.9849  16.312   <2e-16 ***
log(dist)           -1.6449     0.1255 -13.111   <2e-16 ***
cos(angle)           0.1846     0.1081   1.708   0.0894 .
sin(angle)           0.1777     0.1272   1.397   0.1643
geologylimestone.b  -0.1616     0.3549  -0.455   0.6494
geologyother        -0.8520     0.3365  -2.532   0.0122 *
geologytertiary     -0.8835     0.3752  -2.355   0.0196 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error (sqrt(nugget)): 0.4096

Robustness weights:
 All 181 weights are ~= 1.
```

Solution Task 3

```
r.georob.aniso <- georob(log(cu) ~ log(dist) + cos(angle) + sin(angle) + geology,
                  data = d.dornach, locations=~x+y, variogram.model="RMexp",
                  param = c(variance=0.4, nugget=0.05, scale=30),
                  aniso = default.aniso(f1 = 0.3, omega = 50),
                  fit.aniso = default.fit.aniso(f1 = TRUE, omega = TRUE),
                  tuning.psi = 1000)
summary(r.georob)
```

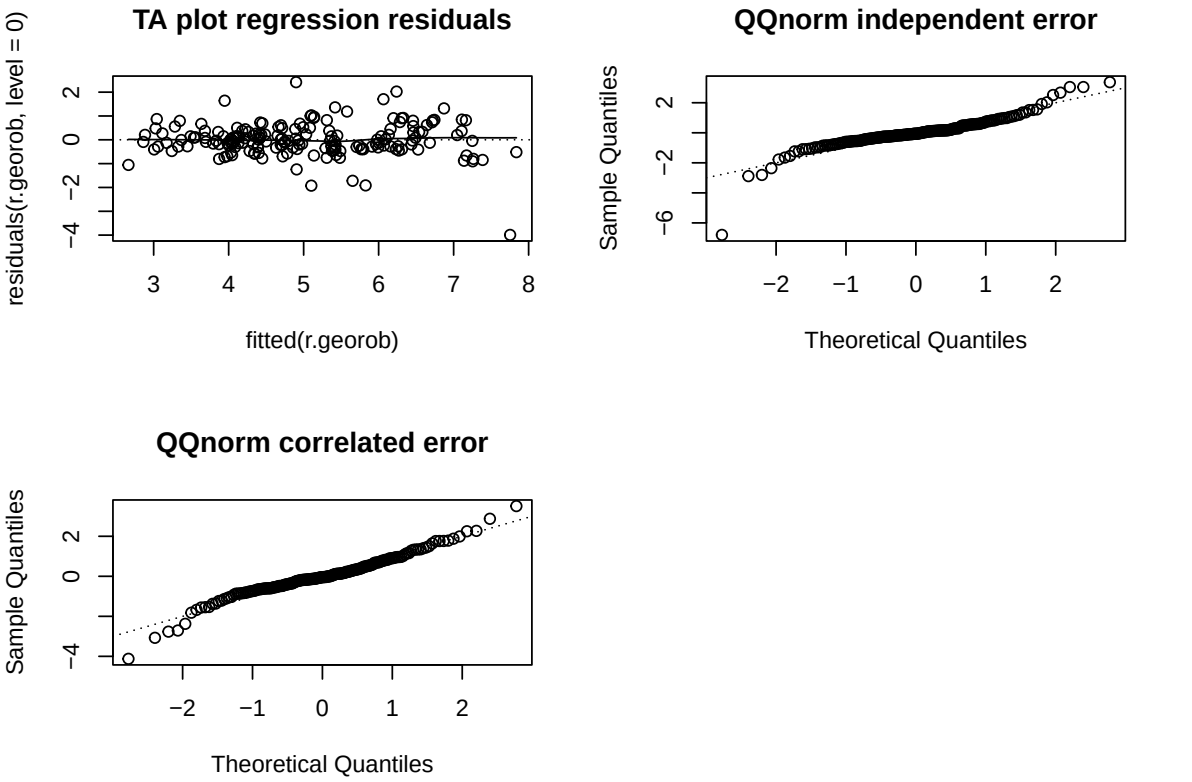
```
Call:georob(formula = log(cu) ~ log(dist) + cos(angle) + sin(angle) +
    geology, data = d.dornach, locations = ~x + y, variogram.model = "RMexp",
    param = c(variance = 0.4, nugget = 0.05, scale = 30), tuning.psi = 1000)

Tuning constant:  1000

Convergence in 8 function and 8 Jacobian/gradient evaluations

Estimating equations (gradient)
```

```
                                  eta         scale
  Gradient           :  3.020576e-04 -6.196804e-03

Maximized restricted log-likelihood: -183.649

Predicted latent variable (B):
     Min       1Q   Median       3Q      Max
-1.92522 -0.26496 -0.02113  0.24321  1.63467

Residuals (epsilon):
     Min       1Q   Median       3Q      Max
-2.06663 -0.10711 -0.01433  0.11256  0.91312

Standardized residuals:
    Min      1Q  Median      3Q     Max
-6.7990 -0.3995 -0.0616  0.4807  3.3698


Gaussian REML estimates

Variogram:  RMexp
               Estimate    Lower   Upper
variance        0.33807  0.21373   0.535
snugget(fixed)  0.00000       NA      NA
nugget          0.16776  0.09349   0.301
scale          58.28482 25.42591 133.609


Fixed effects coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         16.0660     0.9849  16.312   <2e-16 ***
log(dist)           -1.6449     0.1255 -13.111   <2e-16 ***
cos(angle)           0.1846     0.1081   1.708   0.0894 .
sin(angle)           0.1777     0.1272   1.397   0.1643
geologylimestone.b  -0.1616     0.3549  -0.455   0.6494
geologyother        -0.8520     0.3365  -2.532   0.0122 *
geologytertiary     -0.8835     0.3752  -2.355   0.0196 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error (sqrt(nugget)): 0.4096

Robustness weights:
```

```
All 181 weights are ~= 1.
```

Solution Task 4

```
par(mfrow=c(2, 2))
scatter.smooth(fitted(r.georob), residuals(r.georob, level=0),
               main="TA plot regression residuals"); abline(h=0, lty="dotted")
qqnorm(rstandard(r.georob), main="QQnorm independent error")
abline(0, 1, lty="dotted")
qqnorm(ranef(r.georob, standard=TRUE), main="QQnorm correlated error")
abline(0, 1, lty="dotted")
```



Apart from the heavy-tailed distribution of the estimated independent ($\hat{Y}_i$ and spatial auto-correlated error component there are no obvious violations of modelling assumptions apparent. Note that

- the outliers evidently also affect the estimated values of the spatially correlated error component and
- that for both error components the variances are over-estimated because the slope of the curve is less than one except for the tails. Hence, outliers seem to inflate the variance estimates.

This suggests that the spatial model should be estimated robustly. The function `georob()` allows to estimate the parameters of a spatial model robustly, however time of the course was too short to cover this.

1.4 Predictions with random forest

> Task 1
>
> Fit a random forest model with `ranger()` using all available covariates. Since `cu` is strongly positively skewed use `log(cu)`. Maybe plot importance of covariates.

Solution Task 1

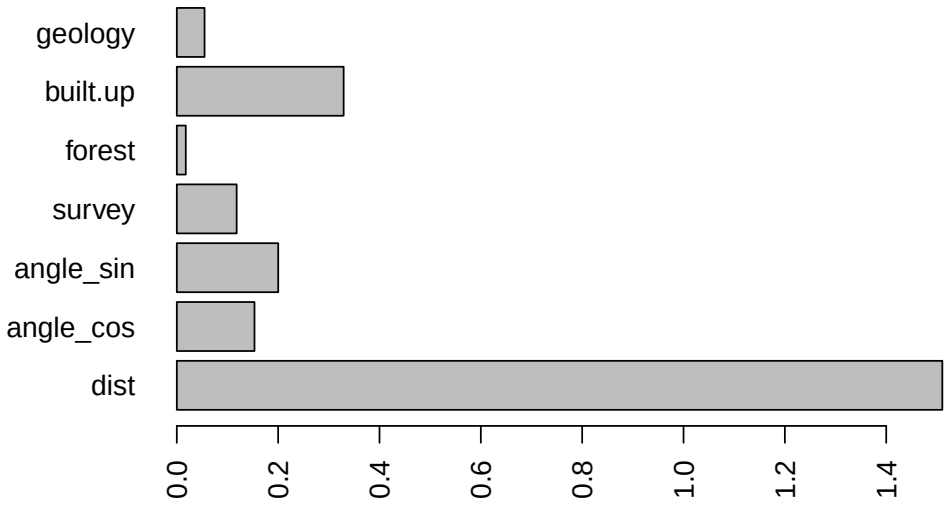
```
library(ranger)
d.dornach$angle_cos <- cos(d.dornach$angle)
d.dornach$angle_sin <- sin(d.dornach$angle)
r.rf <- ranger( log(cu) ~ dist + angle_cos + angle_sin + survey + forest +
                  built.up + geology,
                d.dornach, importance = "permutation")
r.rf
```

```
Ranger result

Call:
 ranger(log(cu) ~ dist + angle_cos + angle_sin + survey + forest +      built.up + geology, d

Type:                             Regression
Number of trees:                  500
Sample size:                      181
Number of independent variables:  7
Mtry:                             2
Target node size:                 5
Variable importance mode:         permutation
Splitrule:                        variance
```

```
OOB prediction error (MSE):       0.4539877
R squared (OOB):                  0.7418209
```

Given the out-of-bag R^2 the model seems to fit the data well.

```
par(oma=c(2,5,1,1))
barplot(importance(r.rf), horiz = T, las = 2)
```



Distance to the smelter seems to be again a very strong predictor. The built up area did not seem relevant in the context of the linear model. Most likely, there are interactions with this factor present, because the tree based structure of random forest is able to capture interactions by subsequent tree splits.

1.5 Model assessment

Task 1

Compute cross-validation for the geostatistical models fited with REML above (isotropic and anisotropic). Use the function `cv()` with `lgn = TRUE` to directly include unbiased backtransformation of the lognormal response.
Would you use `method = "random"` or `method = "block"` for spatial cross-validation?

Task 2

Compute cross-validation for random forest model. Use the same cross-validation subsets as above. You can access them from the `georob` cross-validation object by `yourObject$pred$subset`.

Task 3

Compute scatterplots with predicted vs. observed. Add a lowess smoother and a 1:1-line. Create these plots for the log-transformed results and the back-transformed results.
For random forest use backtransformation without bias correction.

Task 4

Compute meaningful validation metrics and compare the model performance.

Solution Task 1

We do not aim at evaluating spatial extrapolation. We would like to evaluate the model performance within the study area. Therefore, random splitting of the data is sufficient. Leave-block-out cross-validation would yield likely too pessimistic results for the mentioned goal.

Make sure the same subsets are used, either by keeping the seed the same or handing over the subsets with the argument `subset`.

```
r.cv.iso <- cv(r.georob, seed = 13, method = "random", lgn = T)
r.cv.aniso <- cv(r.georob.aniso, subset = r.cv.iso$subset, lgn = T)
```

Solution Task 2

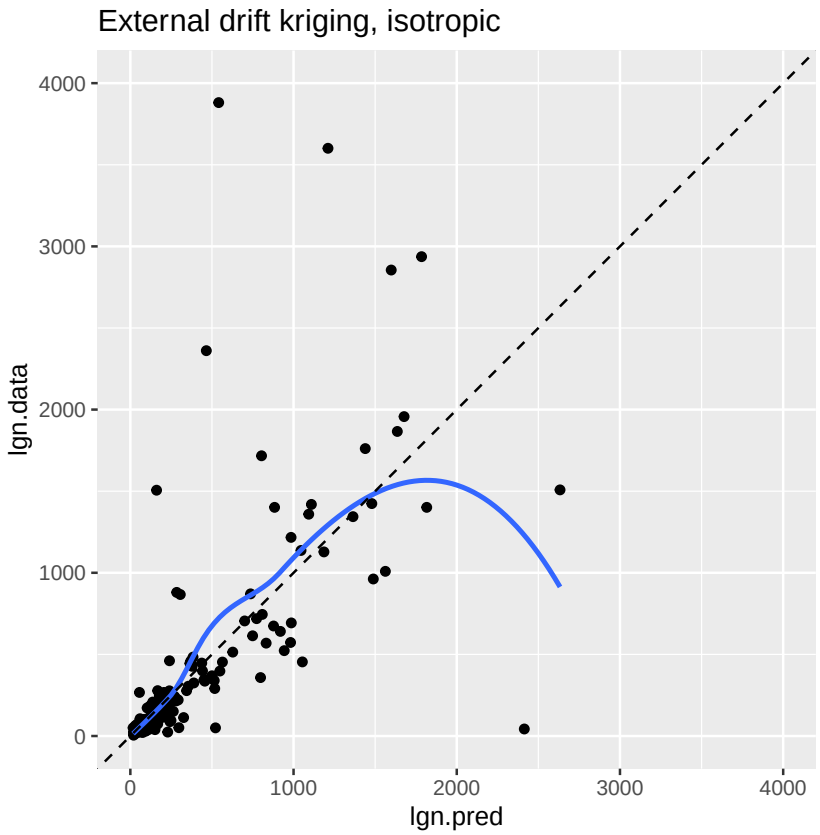
We could use R packages that provide cross-validation functionality like `caret` (indices of subsets can be handed over by using `control()`). But, since the task is simple and we want to make sure that truly the same subsets are used, we quickly implement a `for` loop.

Note: `for` loops are simple but inefficient in R, we would better use an `apply()` function, e.g. `mclapply()` or `foreach()` that allow for paralell computing.

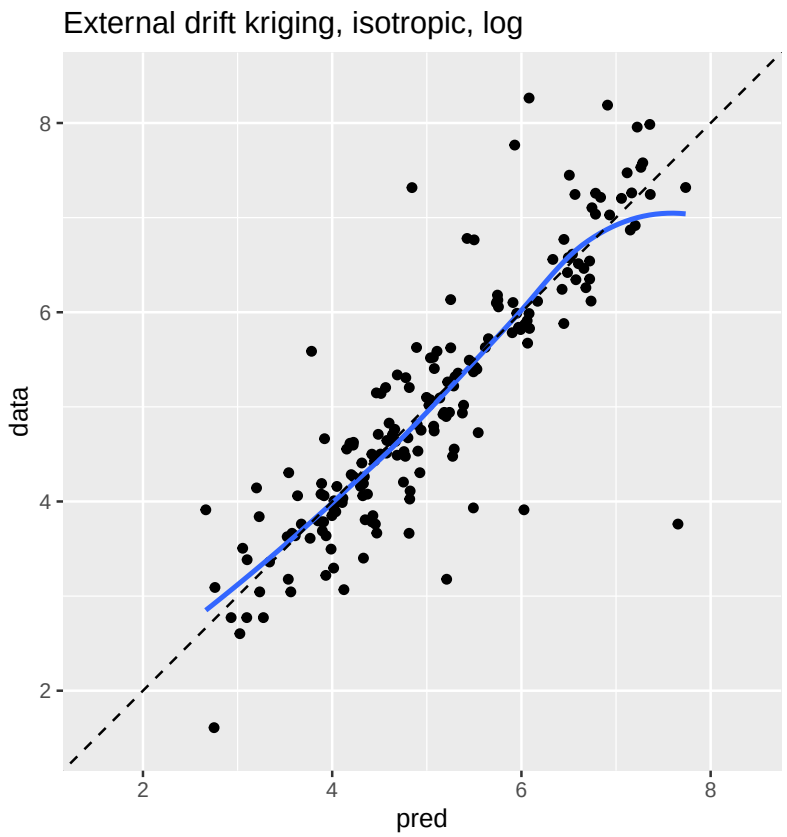
```
d.cv.rf <- d.dornach[, c("x", "y", "cu")]
d.cv.rf$pred <- NA
for( set in 1:10){
   sel.out <- r.cv.iso$pred$subset == set
   rf.set <-  ranger(log(cu) ~ dist + angle_cos + angle_sin + survey + forest +
                       built.up + geology,
                     d.dornach[!sel.out, ])
   d.cv.rf$pred[ sel.out ] <- predict(rf.set, d.dornach[sel.out, ])$predictions
}
```

Solution Task 3

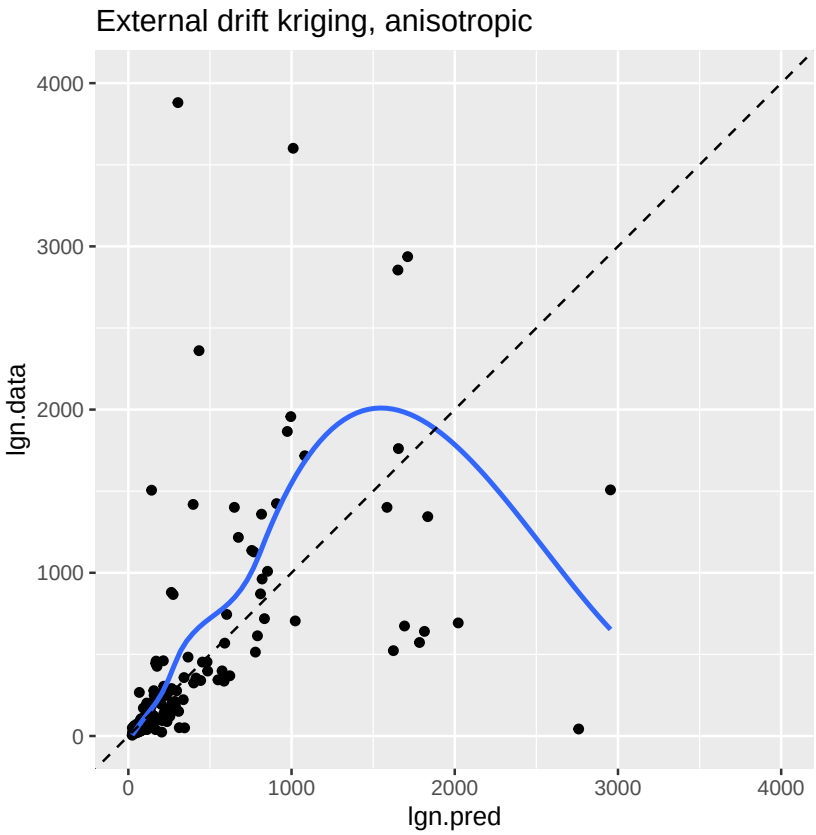
```
library(ggplot2)
par(mfrow = c(2,3))

lim <- c(0,4000)
lim.log <- c(1.5,8.4)

ggplot(r.cv.iso$pred, aes(x = lgn.pred, y = lgn.data)) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) + ylim(lim) + xlim(lim) +
  ggtitle("External drift kriging, isotropic")
```



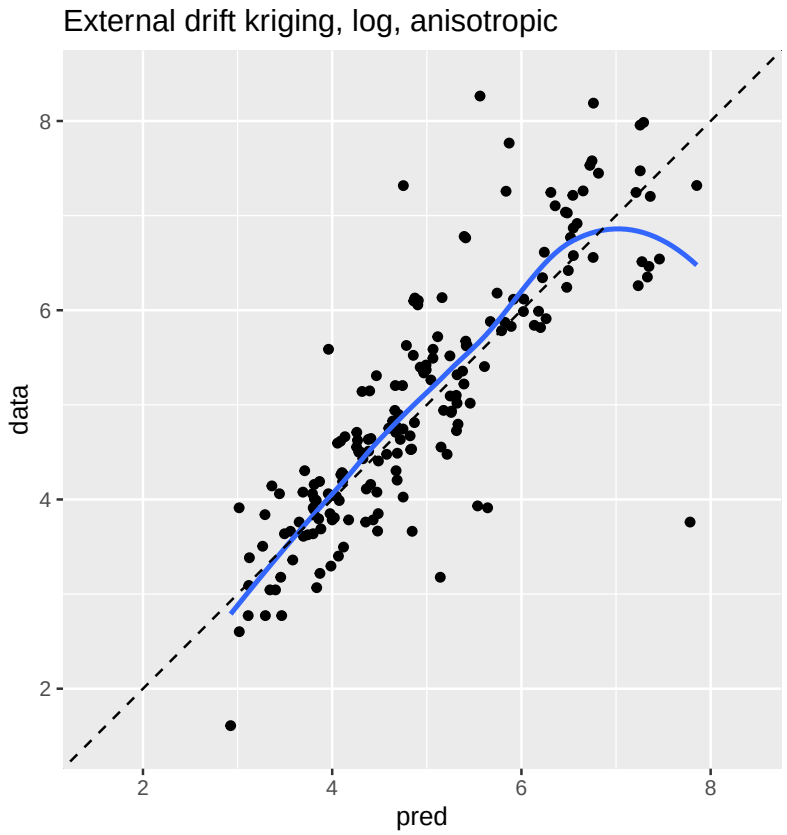

```
ggplot(r.cv.iso$pred, aes(x = pred, y = data)) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) + ylim(lim.log) + xlim(lim.log) +
  ggtitle("External drift kriging, isotropic, log")
```



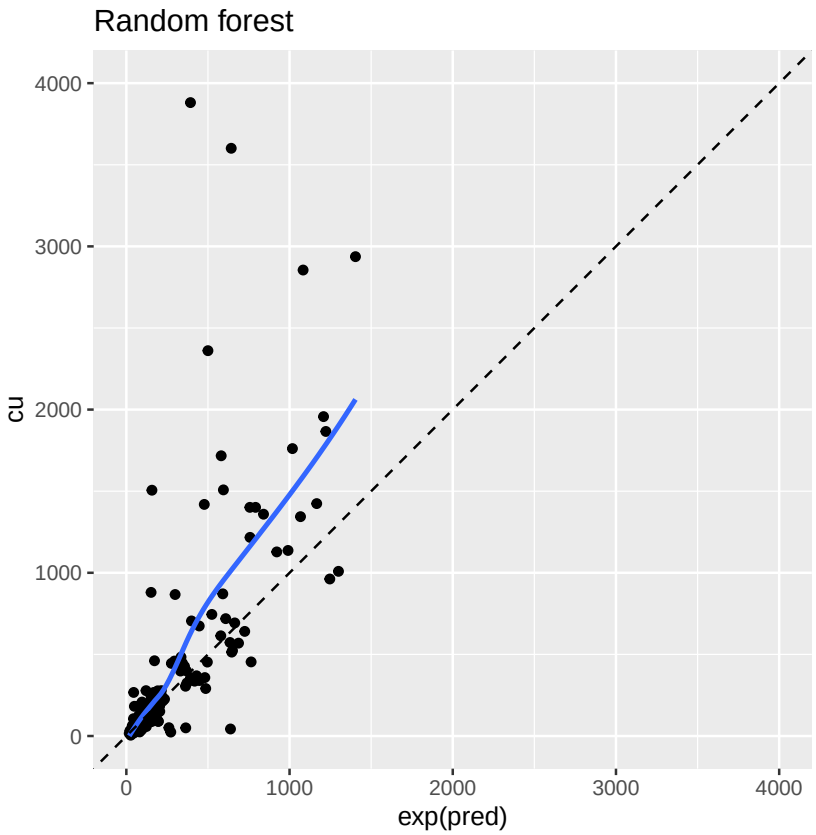

```
ggplot(r.cv.aniso$pred, aes(x = lgn.pred, y = lgn.data)) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) +  ylim(lim) + xlim(lim) +
  ggtitle("External drift kriging, anisotropic")
```



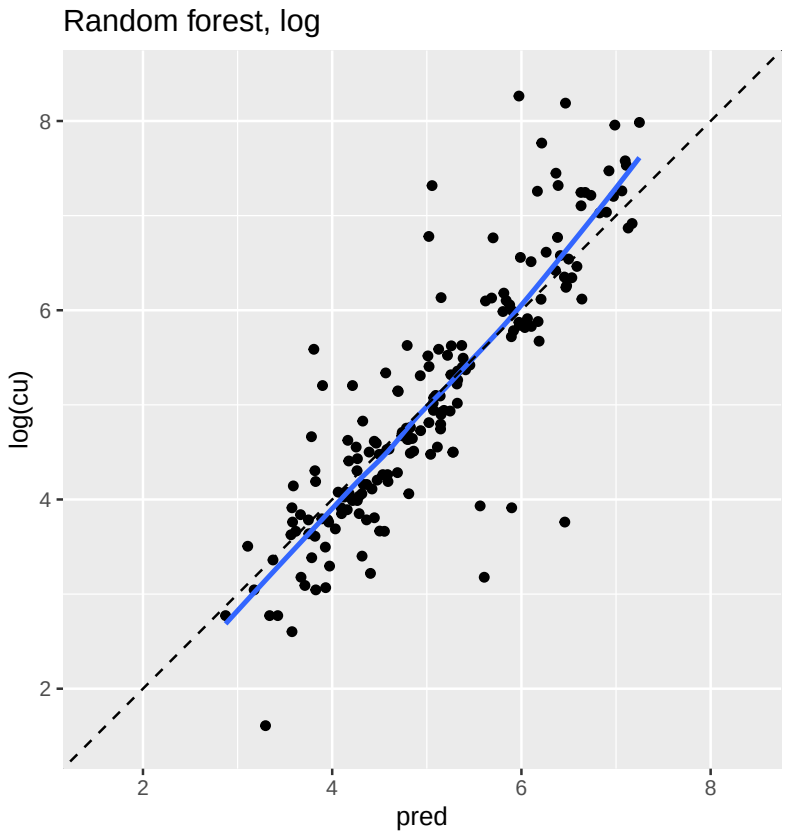

```
ggplot(r.cv.aniso$pred, aes(x = pred, y = data)) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) + ylim(lim.log) + xlim(lim.log) +
  ggtitle("External drift kriging, log, anisotropic")
```




```
ggplot(d.cv.rf, aes(x = exp(pred), y = cu)) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) + ylim(lim) + xlim(lim) +
  ggtitle("Random forest")
```




```
ggplot(d.cv.rf, aes(x = pred, y = log(cu))) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) + ylim(lim.log) + xlim(lim.log) +
  ggtitle("Random forest, log")
```



On log-scale we do not observe conditional bias (bias depending on the predicted value). Overall, there are some outliers, but most data-pairs follow well the 1:1-line.

Backtransformed on the original scale of the response, we observe large poorly predicted outliers. Depending on the model (iso- or anisotropic, RF) the structure of those outliers seems to be different. Especially for RF we observe a "cut-off". There are no values predicted larger than 1500 although there are quite some observations with larger concentrations.

Solution Task 4

Define a formula. We could also use `georob::validate.predictions()`, but there is no R^2 and it requires a standard error.

```
f.metrics <- function(data, pred){
  e <- data - pred
  c( me = mean(e),
     rmse = sqrt(mean(e^2)),
     r2 = 1 - sum(e^2) / sum((data - mean(data))^2)
```

```
    )
}
```

Compute metrics on the log transformed observations and predictions:

```
f.metrics(r.cv.iso$pred$data, r.cv.iso$pred$pred)
```

```
         me        rmse          r2
-0.01180025  0.66041442  0.75058870
```

```
f.metrics(r.cv.aniso$pred$data, r.cv.aniso$pred$pred)
```

```
        me       rmse         r2
0.07223092 0.72345659 0.70069913
```

```
f.metrics(log(d.cv.rf$cu), d.cv.rf$pred)
```

```
          me         rmse          r2
-0.002998657  0.657520549  0.752769709
```

Compute metrics on the backtransformed:

```
f.metrics(r.cv.iso$pred$lgn.data, r.cv.iso$pred$lgn.pred)
```

```
         me        rmse          r2
 16.6647681 453.7127669   0.4759478
```

```
f.metrics(r.cv.aniso$pred$lgn.data, r.cv.aniso$pred$lgn.pred)
```

```
         me        rmse          r2
 33.4686880 534.4724215   0.2727845
```

```
f.metrics(d.cv.rf$cu, exp(d.cv.rf$pred))
```

```
        me       rmse         r2
116.171336 471.590707   0.433835
```

On log-scale, we do not observe problems with marginal bias (me). Regarding RMSE and R^2 the anisotropi model performs worst. Isotropic universal kriging and random forest are in a similar range, while the latter seems to be a tiny bit better.

However, do not over-interpret this small difference. If we would run the cross-validation again with a different random splitting (as you most likely have done), then we would receive slightly different metrics. Given the cross-validation metrics, the two methods are likely similar, i.e. in the range of variation of cross-validation.

For the back-transformed predictions, conclusions are slighly different. The anisotropic model still performs worst given RMSE and R^2. But, random forest now exhibits a large bias, 10 times larger than isotropic universal kriging. This might be due to the missing bias correction for backtransforming the data, but also due to the lack of predicting large values by random forest.

Bias is usually a problem for end-user applications. Therefore, on the back-transformed data we would clearly prefer the isotropic kriging.

1.6 Prediction for `dornach_grid`

Task 1

Compute kriging predictions with the best performing REML fit above. For this use `predict(..., control=control.predict.georob(extended.output=TRUE)`, then use the function `lgnpp()` to obtain the unbiased backtransformation of the lognormal predictions.

Task 2

Compute random forest predictions. Backtransform by `exp()`.

Task 3

Create maps of predictions and kriging standard errors. Compare the maps.

Note: for random forest we could compute the full predictive distribution using `quantreg = T` for the model fit and `predict(..., type = "quantiles", quantiles = ...)`. From this distribution, we could form the required quantity at each pixel.

Solution Task 1

```
library(terra)
d.dornach.grid <- read.table("data/dornach_grid.txt", header=TRUE,
  stringsAsFactors = TRUE)

# add same columns as for model calibration
d.dornach.grid$dist <- with(d.dornach.grid, sqrt(x^2 + y^2))
d.dornach.grid$angle <- with(d.dornach.grid, atan2(y,x))
d.dornach.grid$angle_cos <- cos(d.dornach.grid$angle)
d.dornach.grid$angle_sin <- sin(d.dornach.grid$angle)

# compute lognormal external drift kriging predictions
r.uk <- predict(r.georob, newdata=d.dornach.grid,
                control=control.predict.georob(extended.output=TRUE))

# back-transform prediction results to original scale of measurements
r.uk <- lgnpp(r.uk)
r.uk <- rast(r.uk)
r.uk
```

```
class       : SpatRaster
dimensions  : 251, 301, 12  (nrow, ncol, nlyr)
resolution  : 8, 8  (x, y)
extent      : -1044, 1364, -884, 1124  (xmin, xmax, ymin, ymax)
coord. ref. :
source(s)   : memory
names       :     pred,        se,    lower,    upper,    trend,   var.pred, ...
min values  : 3.087958, 0.2245505, 1.883039, 4.121587, 3.087427, 0.01911292, ...
max values  : 7.976808, 0.6636982, 7.109685, 9.045084, 8.657005, 0.29057960, ...
```

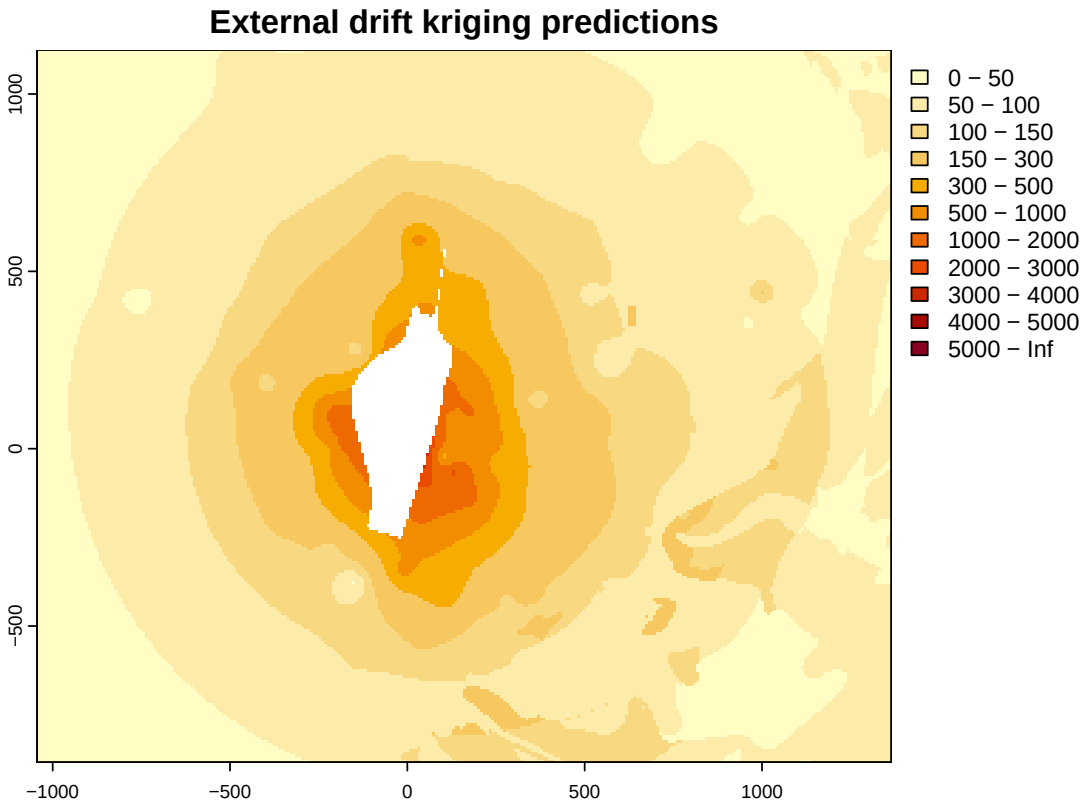
Solution Task 2

```
# remove missing data
t.sel <- complete.cases(d.dornach.grid)
r.rfpred <- predict(r.rf, data = d.dornach.grid[ t.sel, ])

d.dornach.grid$pred.rf[t.sel] <- exp(r.rfpred$predictions)

r.dornach.grid <- rast(d.dornach.grid[,c("x", "y", "pred.rf")])
r.dornach.grid
```

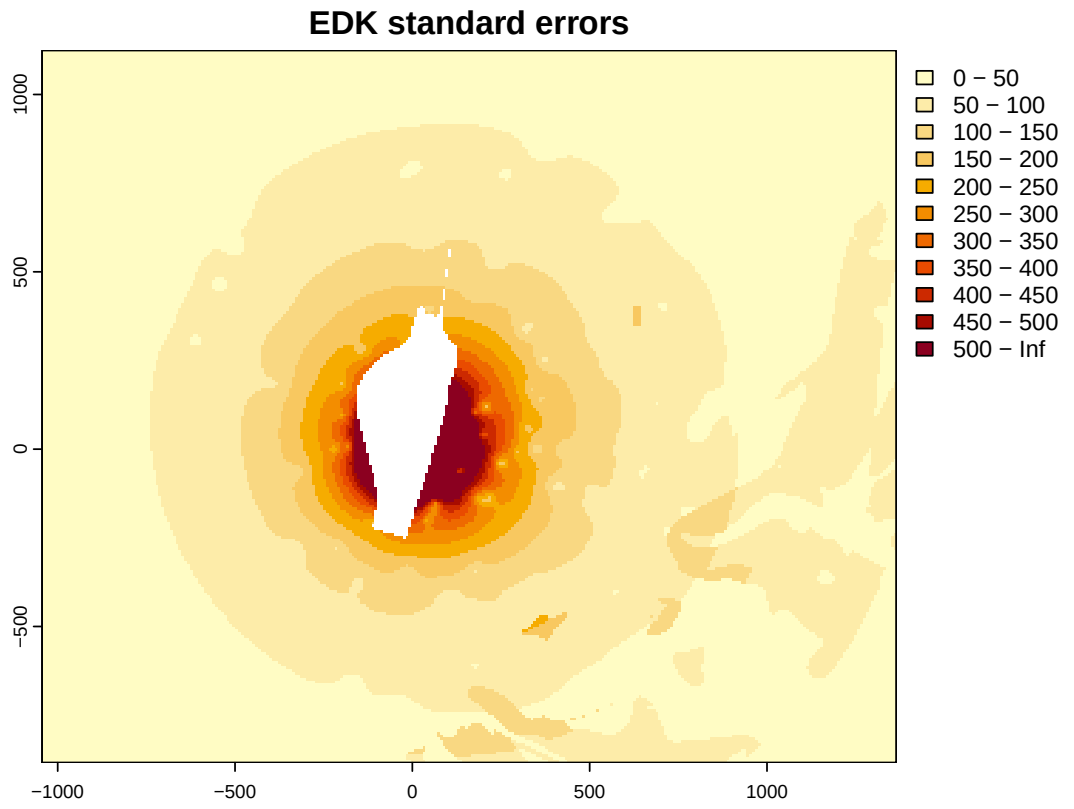
```
class       : SpatRaster
dimensions  : 251, 301, 1  (nrow, ncol, nlyr)
resolution  : 8, 8  (x, y)
extent      : -1044, 1364, -884, 1124  (xmin, xmax, ymin, ymax)
coord. ref. :
source(s)   : memory
name        :    pred.rf
min value   :   49.83535
max value   : 1565.77216
```

Solution Task 3

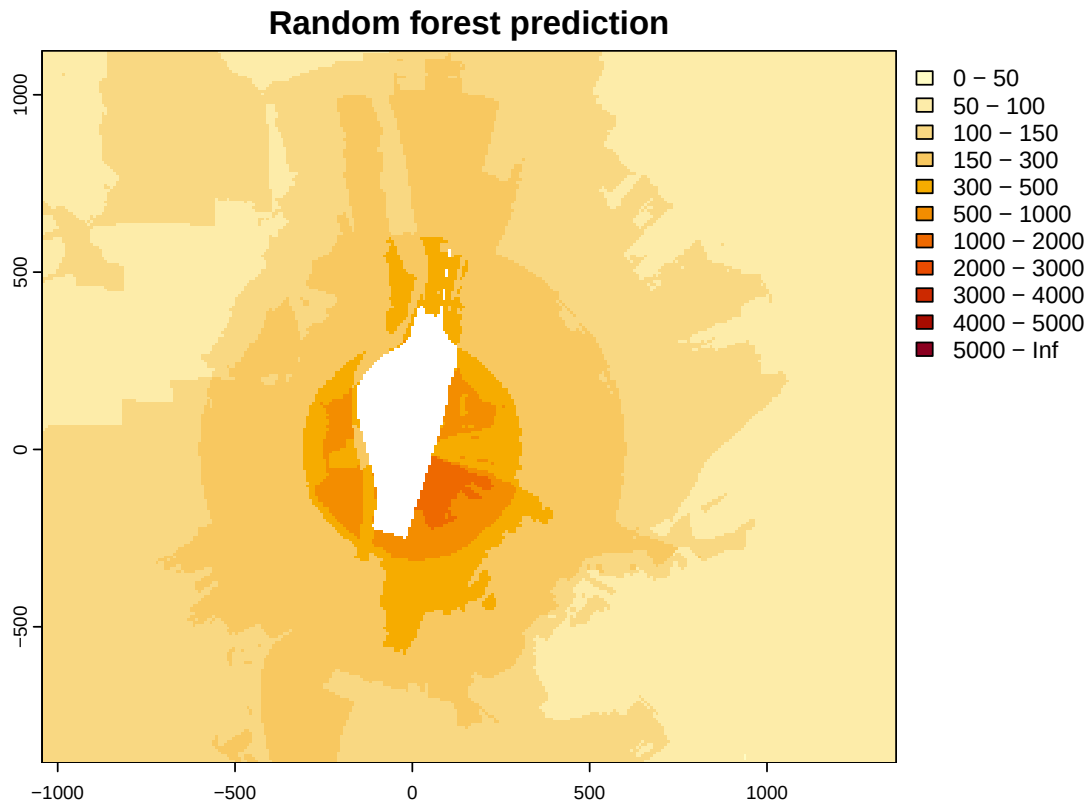
```
plot(r.uk["lgn.pred"], col= hcl.colors(palette = "YlOrRd", 30)[30:1],
     breaks=c(0, 50, 100, 150, 300, 500, 1000, 2000, 3000, 4000, 5000, Inf),
     main = "External drift kriging predictions")
```



```
plot(r.uk["lgn.se"], col= hcl.colors(palette = "YlOrRd", 30)[30:1],
     breaks = c(seq(0, 500, by=50), Inf),
     main = "EDK standard errors")
```




```
plot(r.dornach.grid["pred.rf"], col= hcl.colors(palette = "YlOrRd", 30)[30:1],
     breaks=c(0, 50, 100, 150, 300, 500, 1000, 2000, 3000, 4000, 5000, Inf),
     main = "Random forest prediction")
```



The spatial pattern predicted by random forest looks considerably different than predicted by isotropic universal kriging. Again, we do not see the large concentrations being predicted around the smelter. In addition, the mapped `cu` surface is rather bumpy and seems dissected by the regression tree structure of random forest.

We would now need to investigate the feasibility of the mapped pattern with the original data and by checking with the situation of the smelter (location in valley, forested areas etc.). Moreover, we could discuss the result with a pollution expert to get more insight on the expected true pattern.

For random forest it is usually necessary to have a larger number of predictors to create smooth prediction surfaces. Therefore, we could also investigate if there are more relevant predictors for the study area.

2 Model assessment pH predictions Berne data set

In lab session 3 you computed predictions for topsoil pH in the Berne study area. We did not yet fully inspect the model performance.

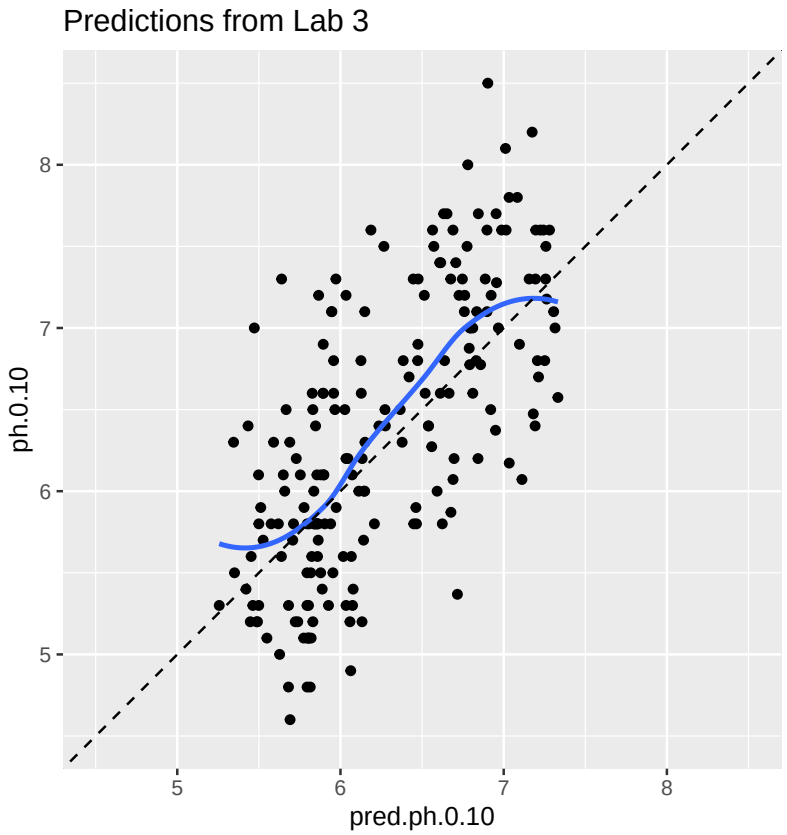
Find your saved CSV file from session 3 or download the example CSV. CSV

> Task 1
>
> Create plots and metrics as you see fit. What is the model performance? Are there relevant problems?

Solution Task 1

```
d.val.berne <- read.csv("data/lab3-validation-set-ph-predictions.csv")

ggplot(d.val.berne, aes(x = pred.ph.0.10, y = ph.0.10)) +
  geom_point() + geom_smooth(se=F) +
  geom_abline (slope=1, intercept = 0, linetype = "dashed") +
  coord_fixed(ratio = 1) + ylim(c(4.5,8.5)) + xlim(c(4.5,8.5)) +
  ggtitle("Predictions from Lab 3")
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




```
f.metrics(d.val.berne$ph.0.10, d.val.berne$pred.ph.0.10)
```

```
       me      rmse        r2
0.1015193 0.6183406 0.4519854
```

According to the help page of the `berne` data the `validation` data set was formed by data-splitting It was assigned at random by using weights to ensure even spatial coverage of the agricultural area.

Therefore, we get a decent estimate of the predictive performance of the model. If you check the point distribution, however, they are somewhat clustered and there are areas without any samples. We would now need to investigate if those "empty" areas are expected to be different. This could be done by checking the covariate value distributions and/or by discussing it with a soil surveyor that knows the area.

Regarding the results: We do not observe a marginal bias, however, it seems we have a slight conditional bias with values between 6–7 being under-predicted. In addition, largest and

smallest values do not get predicted, it seems random forest smoothes the value range of the predictions compared to the value range of the observed response.

The RMSE, i.e. the standard deviation of the errors, is a bit more than half a pH unit (0.62). The R^2 is only medium. Soil mapping studies, however, have often R^2 in this range as variability of soils is large, often not enough samples are available to describe the full "diversity" and measurement error are considerable.