Areal Data and Proximity

HES 505 Fall 2024: Session 19

Carolyn Koehn

Objectives

By the end of today you should be able to:

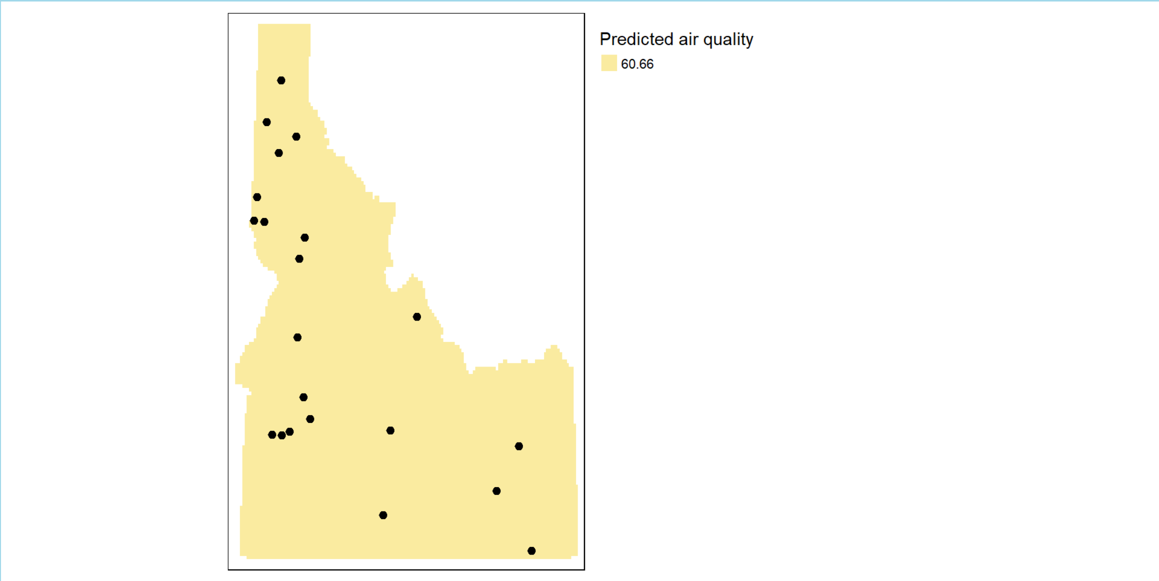
Statistical Interpolation

Statistical Interpolation

Trend Surface Modeling

- Basically a regression on the coordinates of your data points
- Coefficients apply to the coordinates and their interaction
- Relies on different functional forms

0th Order Trend Surface



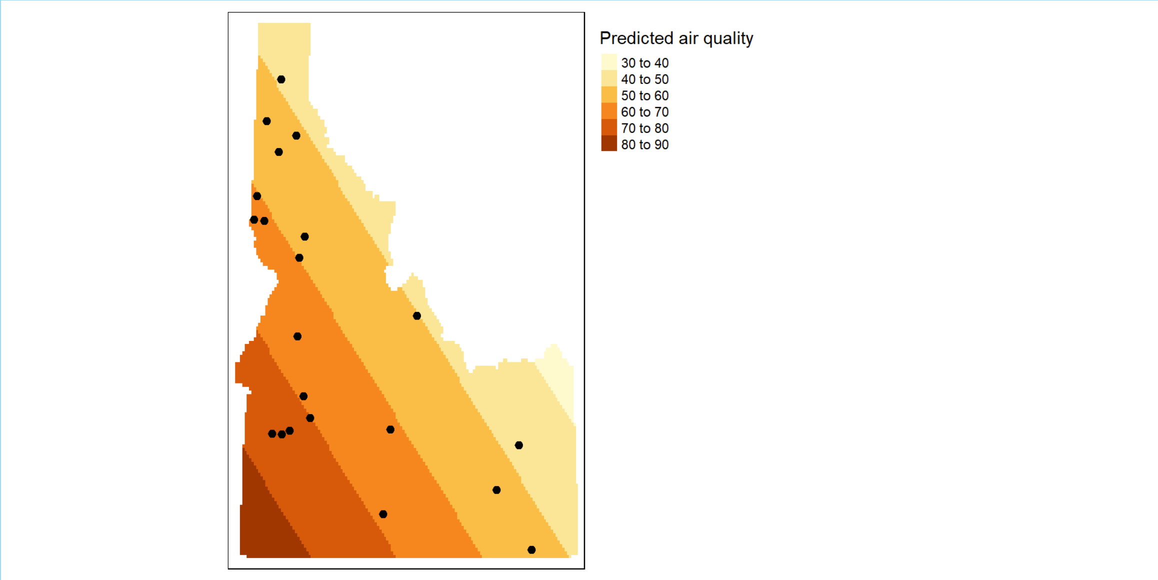
- Simplest form of trend surface
- $Z = a$ where a is the mean value of air quality
- Result is a simple horizontal surface where all values are the same.

0th order trend surface

```
#set up interpolation grid
# Create an empty grid where n is the total number of cells
bbox <- st_bbox(id.cty)
grd <- st_make_grid(id.cty, n=150,
                    what = "centers") %>%
  st_as_sf() %>%
  mutate(X = st_coordinates(.)[, 1],
         Y = st_coordinates(.)[, 2])

# Define the polynomial equation
f.0  <- as.formula(meanpm25 ~ 1)

# Run the regression model
lm.0 <- lm( f.0 , data=aq.sum)

# Use the regression model output to interpolate the surface
grd$var0.pred <- predict(lm.0, newdata = grd)
# Use data.frame without geometry to convert to raster
```

1st Order Trend Surface

- Creates a slanted surface
- $Z = a + bX + cY$
- X and Y are the coordinate pairs

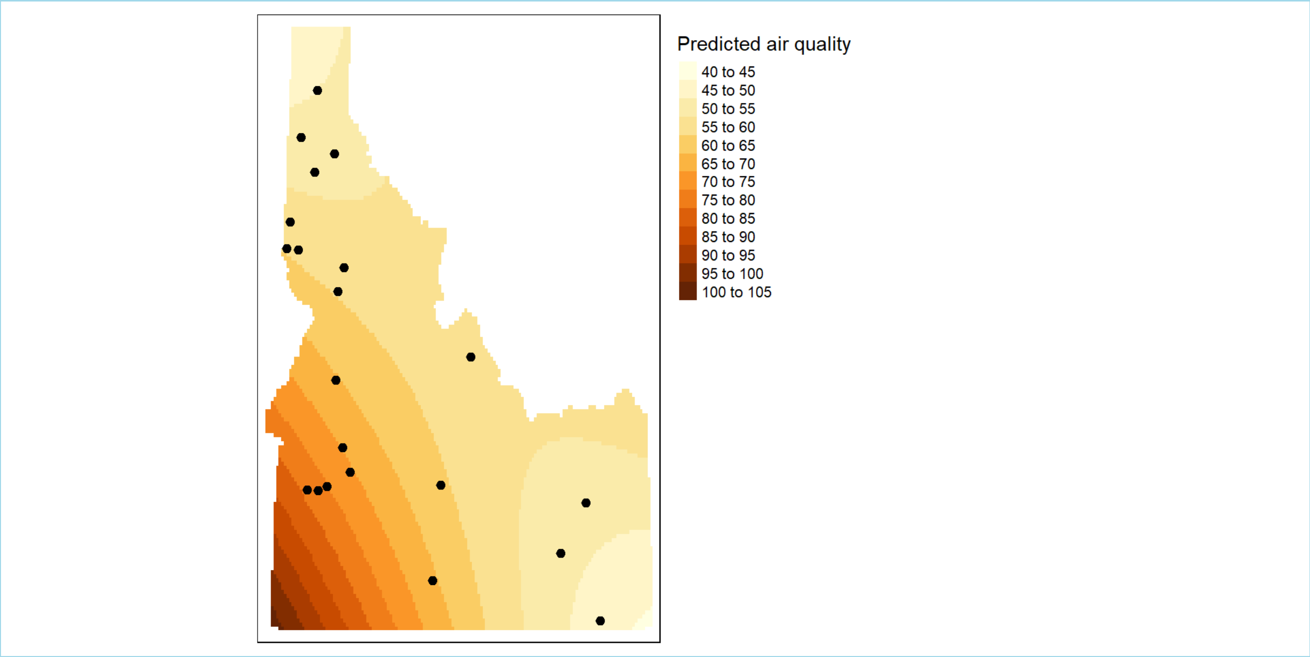


1st Order Trend Surface

```r
# Define the polynomial equation
f.1  <- as.formula(meanpm25 ~ X + Y)

aq.sum$X <- st_coordinates(aq.sum)[,1]
aq.sum$Y <- st_coordinates(aq.sum)[,2]

# Run the regression model
lm.1 <- lm( f.1 , data=aq.sum)

# Use the regression model output to interpolate the surface
grd$var1.pred <- predict(lm.1, newdata = grd)
# Use data.frame without geometry to convert to raster
dat.1st <- grd %>%
  select(X, Y, var1.pred) %>%
  st_drop_geometry()

# Convert to raster object to take advantage of rasterVis' imaging
# environment
```

2nd Order Trend Surfaces

- Produces a parabolic surface
- $Z = a + bX + cY + dX^2 + eY^2 + fXY$
- Highlights the interaction of both directions



2nd Order Trend Surfaces

```r
# Define the 1st order polynomial equation
f.2 <- as.formula(meanpm25 ~ X + Y + I(X*X)+I(Y*Y) + I(X*Y))

# Run the regression model
lm.2 <- lm( f.2, data=aq.sum)

# Use the regression model output to interpolate the surface
grd$var2.pred <- predict(lm.2, newdata = grd)
# Use data.frame without geometry to convert to raster
dat.2nd <- grd %>%
  select(X, Y, var2.pred) %>%
  st_drop_geometry()

r   <- rast(dat.2nd, crs = crs(grd))
r.m <- mask(r, st_as_sf(id.cty))

tm_shape(r.m) + tm_raster(n=10, title="Predicted air quality") +
  tm_shape(aq.sum) +
```

Kriging

- Previous methods predict z as a (weighted) function of distance
- Treat the observations as perfect (no error)
- If we imagine that z is the outcome of some spatial process such that:

$$z(\mathbf{x}) = \mu(\mathbf{x}) + \epsilon(\mathbf{x})$$

then any observed value of z is some function of the process ($\mu(\mathbf{x})$) and some error ($\epsilon(\mathbf{x})$)

- Kriging exploits autocorrelation in $\epsilon(\mathbf{x})$ to identify the trend and interpolate accordingly

Autocorrelation

- **Correlation** the tendency for two variables to be related
- **Autocorrelation** the tendency for observations that are closer (in space or time) to be correlated
- **Positive autocorrelation** neighboring observations have ϵ with the same sign
- **Negative autocorrelation** neighboring observations have ϵ with a different sign (rare in geography)

Ordinary Kriging

- Assumes that the deterministic part of the process ($\mu(\mathbf{x})$) is an unknown constant (μ)

$$z(\mathbf{x}) = \mu + \epsilon(\mathbf{x})$$

Steps for Ordinary Kriging

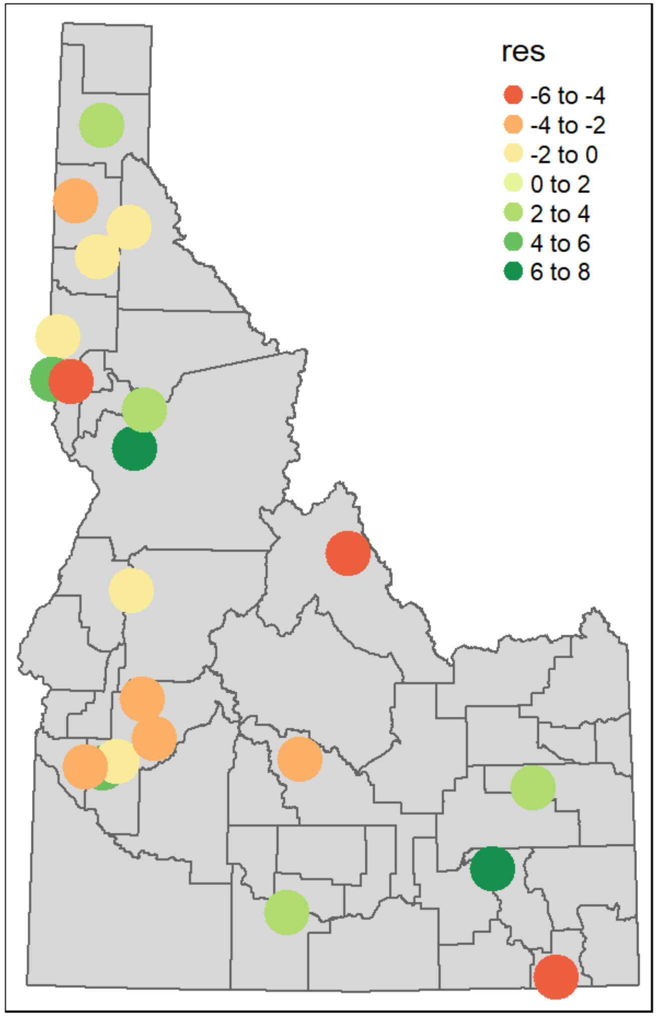
- Removing any **spatial trend** in the data (if present).
- Computing the **experimental variogram**, γ, which is a measure of spatial autocorrelation.
- Defining an **experimental variogram model** that best characterizes the spatial autocorrelation in the data.
- Interpolating the surface using the experimental variogram.
- Adding the kriged interpolated surface to the trend interpolated surface to produce the final output.

Removing Spatial Trend

- Mean and variance need to be constant across study area
- Trend surfaces indicate that is not the case
- Need to remove that trend

```r
f.2 <- as.formula(meanpm25 ~ X + Y + I(X*X)+I(Y*Y) + I(X*Y))

# Run the regression model
lm.2 <- lm( f.2, data=aq.sum)

# Copy the residuals to the point object
aq.sum$res <- lm.2$residuals
```

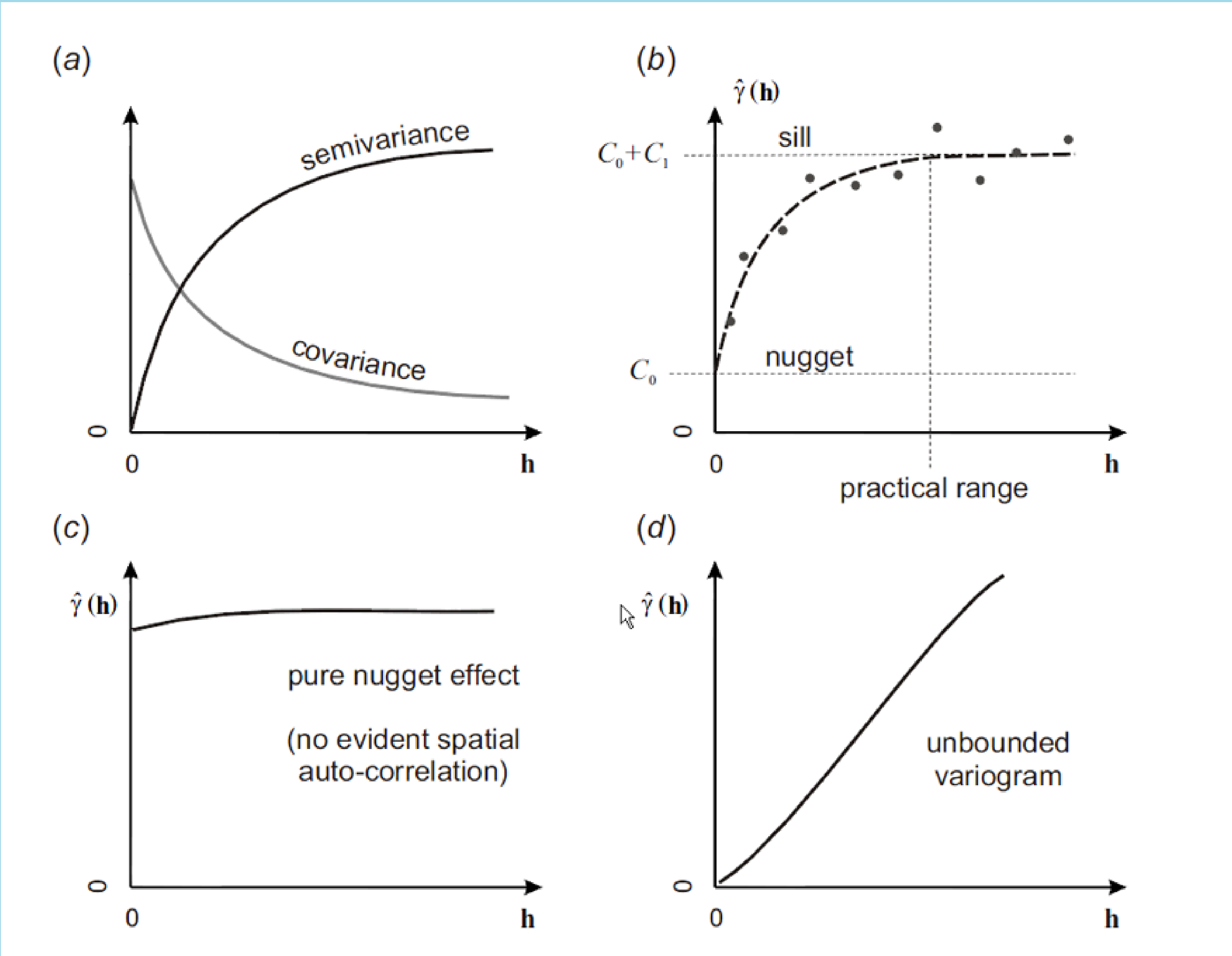
Removing the trend



Calculate the experimental variogram

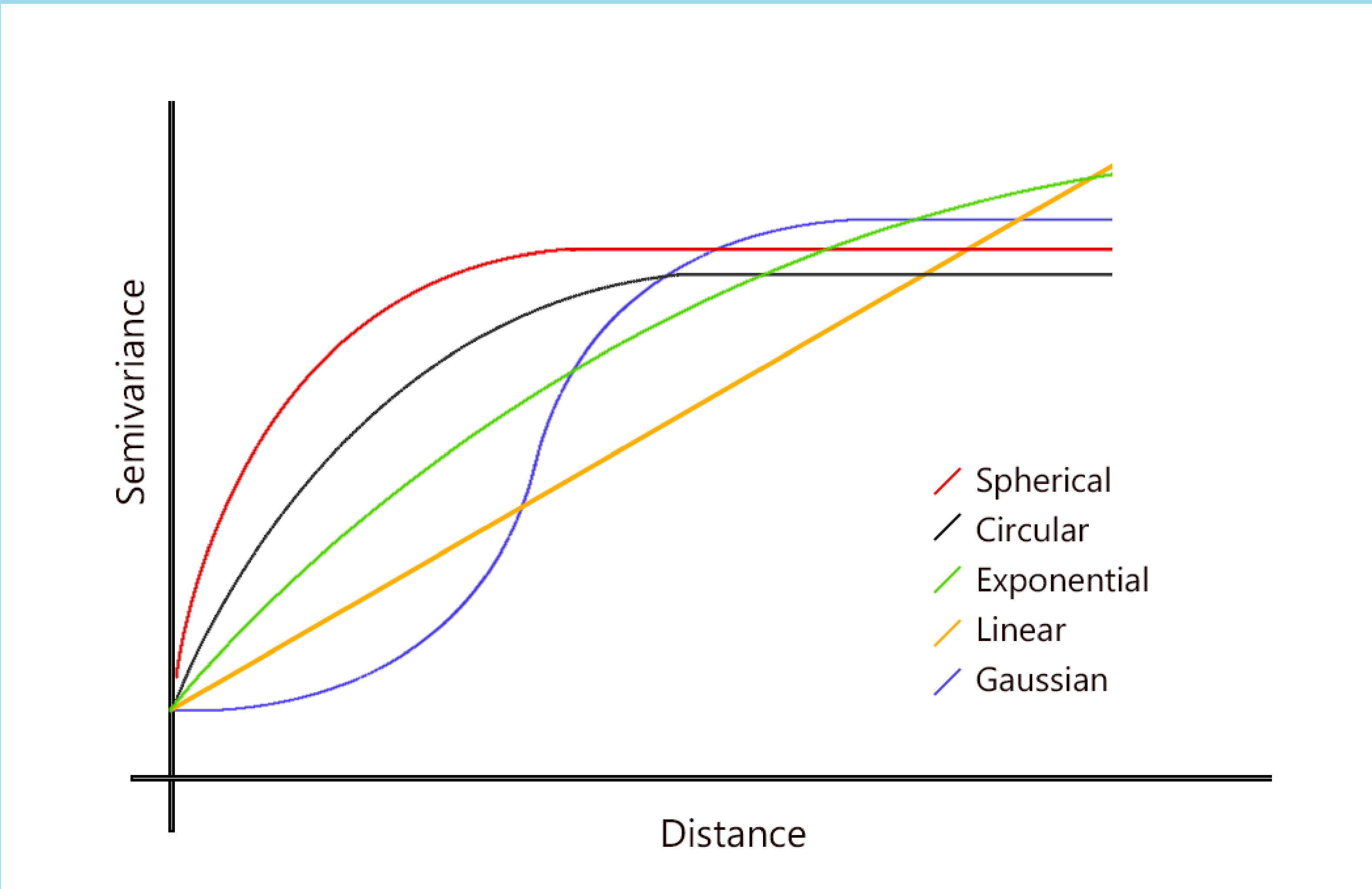
- **nugget** - the proportion of semivariance that occurs at small distances
- **sill** - the maximum semivariance between pairs of observations
- **range** - the distance at which the **sill** occurs
- **experimental** vs. **fitted** variograms

A Note about Semivariograms



Fitted Semivariograms

- Rely on functional forms to model semivariance



Calculate the experimental variogram

```
var.cld  <- gstat::variogram(res ~ 1, aq.sum, cloud = TRUE)
var.df  <- as.data.frame(var.cld)
index1  <- which(with(var.df, left==21 & right==2))
```

Calculate the experimental variogram

```r
OP <- par( mar=c(4,6,1,1))
plot(var.cld$dist/1000 , var.cld$gamma, col="grey",
     xlab = "Distance between point pairs (km)",
     ylab = expression( frac((res[2] - res[1])^2 , 2)) )
```

```r
par(OP)
```

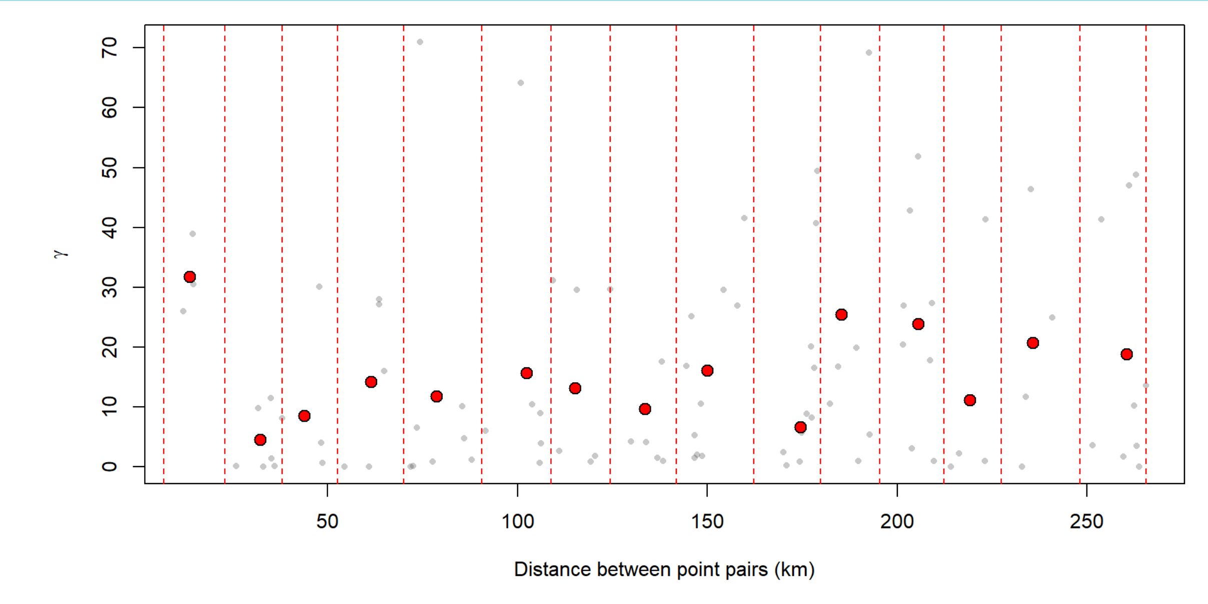
Simplifying the cloud plot

```r
# Compute the sample experimental variogram
var.smpl <- gstat::variogram(f.2, aq.sum, cloud = FALSE)

bins.ct <- c(0, var.smpl$dist , max(var.cld$dist) )
bins <- vector()
for (i in 1: (length(bins.ct) - 1) ){
  bins[i] <- mean(bins.ct[ seq(i,i+1, length.out=2)] )
}
bins[length(bins)] <- max(var.cld$dist)
var.bins <- findInterval(var.cld$dist, bins)
```

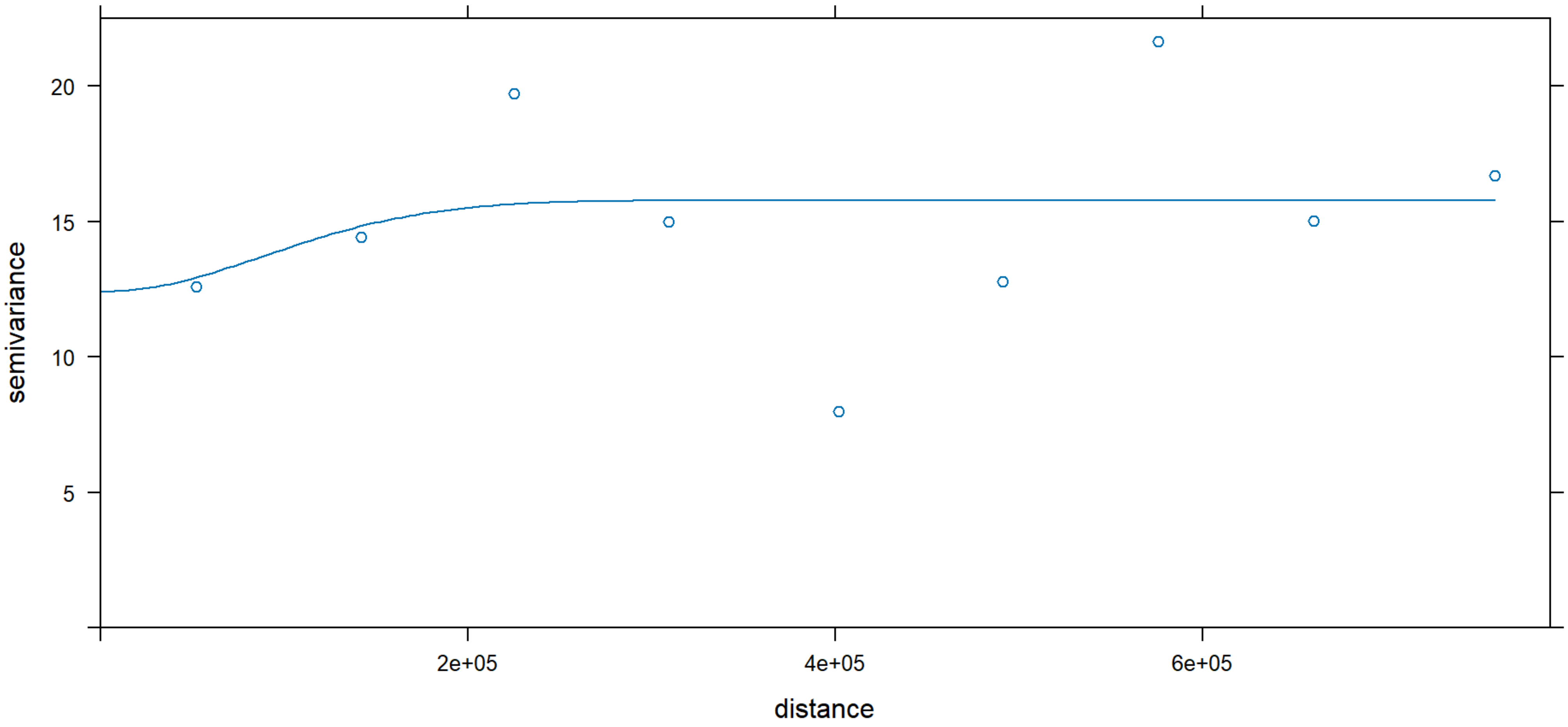
Simplifying the cloud plot

```
# Point data cloud with bin boundaries
OP <- par( mar = c(5,6,1,1))
plot(var.cld$gamma ~ eval(var.cld$dist/1000), col=rgb(0,0,0,0.2), pch=16, c
     xlab = "Distance between point pairs (km)",
     ylab = expression( gamma ) )
points( var.smpl$dist/1000, var.smpl$gamma, pch=21, col="black", bg="red",
abline(v=bins/1000, col="red", lty=2)
```

```
par(OP)
```



Looking at the sample Variogram

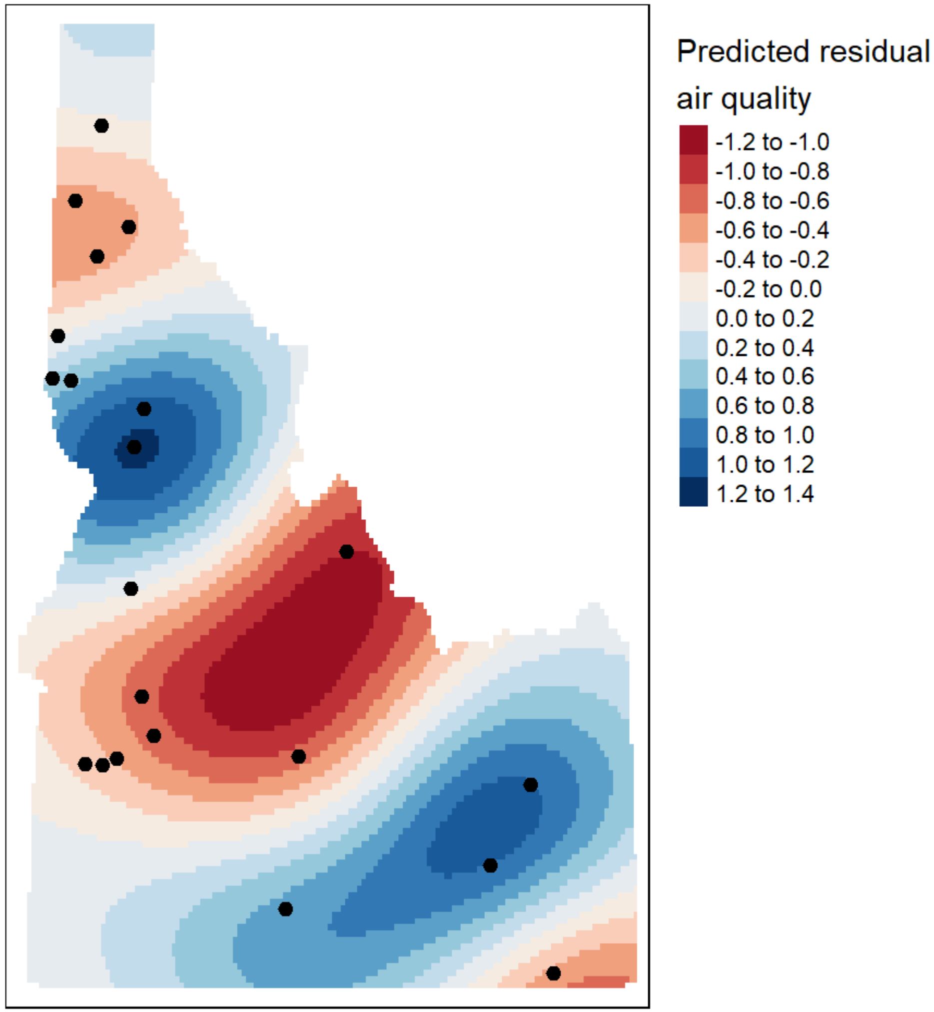


Estimating the sample variogram

```
# Compute the sample variogram, note the f.2 trend model is one of the para
# passed to variogram(). This tells the function to create the variogram on
# the de-trended data
var.smpl <- gstat::variogram(f.2, aq.sum, cloud = FALSE, cutoff = 1000000,


# Compute the variogram model by passing the nugget, sill and range values
# to fit.variogram() via the vgm() function.
dat.fit  <- gstat::fit.variogram(var.smpl, gstat::vgm(nugget = 12, range= 6

# The following plot allows us to gauge the fit
plot(var.smpl, dat.fit)
```

Ordinary Kriging

[using ordinary kriging]

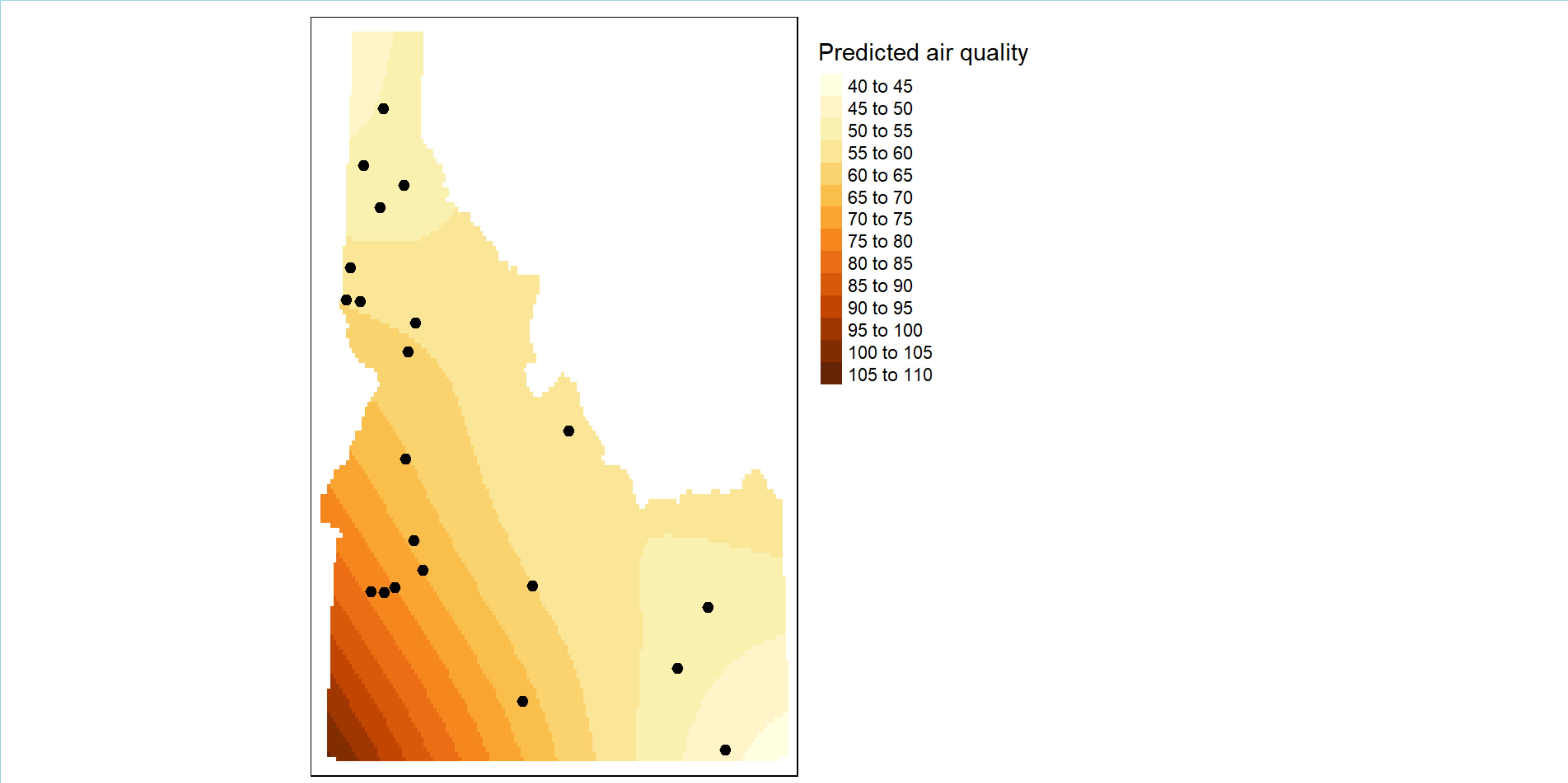


Ordinary Kriging

```r
dat.krg <- gstat::krige( formula = res~1,
                         locations = aq.sum,
                         newdata = grd[, c("X", "Y", "var2.pred")],
                         model = dat.fit)
```

Combining with the trend data

[using universal kriging]



Combining with the trend data

```
dat.krg <- gstat::krige( formula = f.2,
                         locations = aq.sum,
                         newdata = grd[, c("X", "Y", "var2.pred")],
                         model = dat.fit)

dat.krg.preds <-  dat.krg %>%
  mutate(X = st_coordinates(.)[, 1],
         Y = st_coordinates(.)[, 2]) %>%
  select(X, Y, var1.pred) %>%
  st_drop_geometry()

r <- rast(dat.krg.preds, crs = crs(grd))
r.m <- mask(r, st_as_sf(id.cty))

# Plot the raster and the sampled points
tm_shape(r.m) + tm_raster(n=10, title="Predicted air quality") +tm_shape(aq
  tm_legend(legend.outside=TRUE)
```

Visualizing Uncertainty