Nicholas Milikich
ACMS 60855 Spatio-Temporal Statistics for Environmental Applications
April 16, 2020

# Midterm 2

In the 1980's, the Department of Energy's Civilian Radioactive Waste Management Program surveyed a number of potential sites for a new high-level nuclear waste disposal site, one of which was located in the Palo Duro Basin in Deaf Smith county, Texas[1] (Figure 1, left). Radioactive waste containers run the risk of leaking, especially in the corrosive salty environment in which they are stored, and the potential consequences of radioactive waste leakage are especially severe if the leakage has the potential to come into contact with a populated area. The potential nuclear waste site is located about 60 km southwest of Amarillo, Texas, and both locations overlie the Wolfcamp aquifer. This report is concerned with assessing the patterns of water movement within this aquifer to determine whether there is a risk of radioactive waste leakage diffusing towards this nearby populated area.

This will be achieved by estimating a two-dimensional potentiometric surface of pressure within the Wolfcamp aquifer. At 85 scattered locations in the aquifer, the water pressure is measured indirectly by measuring piezometric head, which involves boring a narrow hole and measuring the height above the surface of the aquifer that the water reaches in a piezometer (a narrow tube)[2]; the pressure data are therefore reported in meters. The data have been preprocessed by the contracting company that collected them by deleting depressured and locally overpressured or underpressured data to more accurately represent the potentiometric surface[1]. The potentiometric surface will be estimated by performing kriging on the available data. This surface will be used as an estimate of water flow patterns in the aquifer (by assuming that water diffuses from areas of higher pressure to areas of lower pressure) to determine whether water flows from the potential nuclear waste site in Deaf Smith county towards Amarillo.

As shown in the left panel of Figure S1 (Supplemental Figure 1), the piezometric head data appear to follow a bimodal distribution. Although the distribution appears to differ from Gaussian, the sample size of 85 measurements is rather small, and using larger bins as in the right panel, the data might reasonably be approximated as Gaussian. Additionally, the data do not have an obvious skew, and for these reasons it is doubtful whether a log transformation would be beneficial. (Indeed, a log transformation does not remove the bimodality of the distribution; see appendix, "Log transform.")

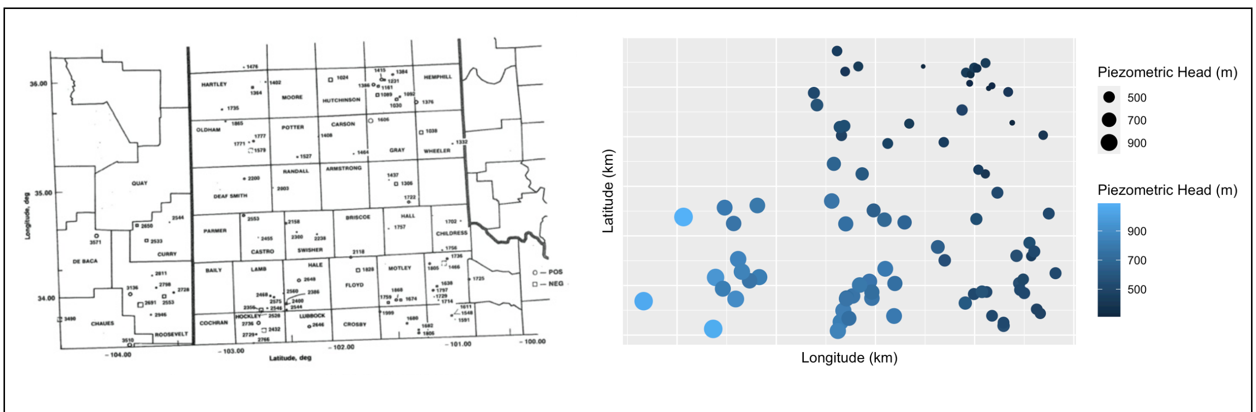



Figure 1: Visualization of the wolfcamp dataset, including the location of the sampled sites in the Texas panhandle and eastern New Mexico (left) and a bubble plot (right). Top left figure reproduced from Harper & Furr, 1986[1].

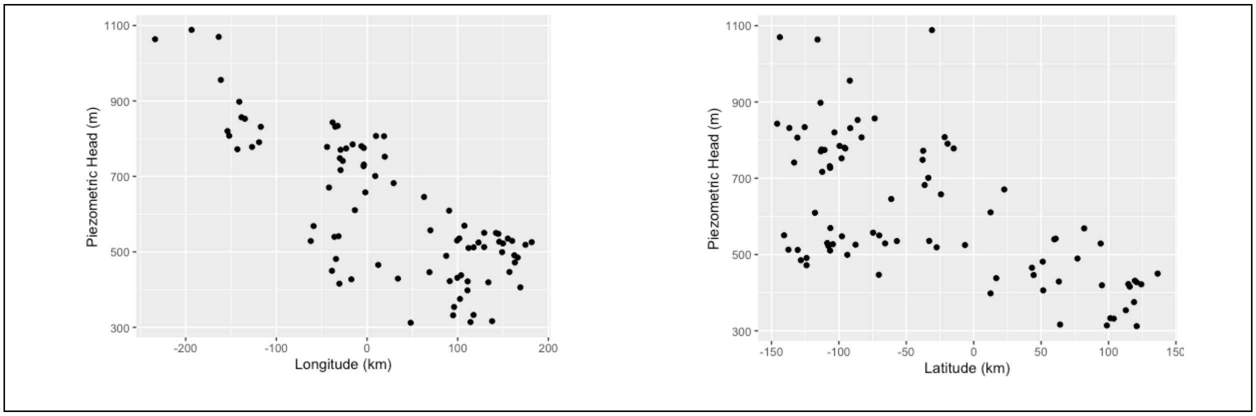


Figure 2: Plot of piezometric head data against longitude (left) and latitude (right).

The data are visualized with their locations on a map and with a bubble plot in Figure 1. A visual inspection of the bubble plot suggests that values of piezometric head are largest in the southwest and smallest in the northeast, and vary somewhat smoothly between. A plot of the piezometric head values against longitude and latitude (Figure 2) confirms this visual intuition. There appears to be a strong negative correlation between piezometric head and both longitude and latitude; piezometric head values tend to decrease when moving north or east. This spatial dependence is fortunate because of the lack of other potential covariates, so the data is detrended by fitting a linear regression on the longitudinal and latitudinal coordinates.

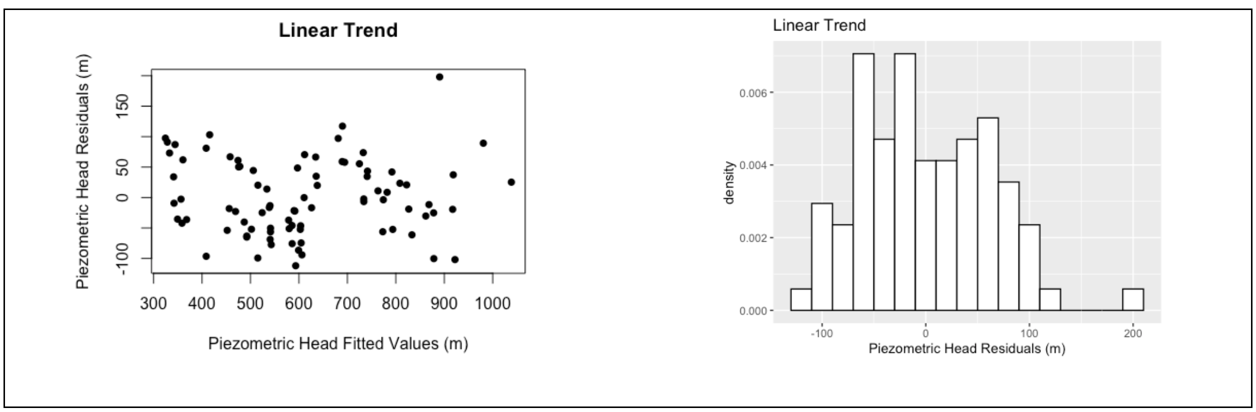


Figure 3: Residual analysis of fitting a linear model for piezometric head on x and y coordinates, including a plot of fitted values against residuals (left) and a histogram of the residuals (right).

Residual analysis upon fitting a linear trend to the data is shown in Figure 3. There are no obvious patterns in the plot of fitted values against residuals on the left, indicating that there is not significant information in the residuals left to be captured. There is, however, an outlier (the point with a fitted value of around 900 m and a residual value of nearly 200 m), which is also apparent in the histogram of residuals on the right. The outlier occurs at site 78, the site with the highest piezometric head value. The histogram suggests two noteworthy features. First, the residuals appear to be somewhat right-skewed. In order to make accurate statistical inferences based on this model, they should follow a Gaussian distribution, so this skewness might cast doubt on the accuracy of any predictions. Performing a log transform on the data does not significantly improve the Gaussianity of the residuals of a linear model based on the linearity of a Q-Q plot (see appendix, “Log-transformed linear model”), so a log transform is not applied. Second, the outlier is very apparent in the histogram. Removing the outlier is not a suitable option: not using all available information to detrend the data simply leads to a larger residual value at this site on which to perform kriging (see appendix, "Linear trend without the outlier"), and thus simply puts off the problem of the outlier to a later part of the analysis instead of addressing it. The outlier is left in the model.

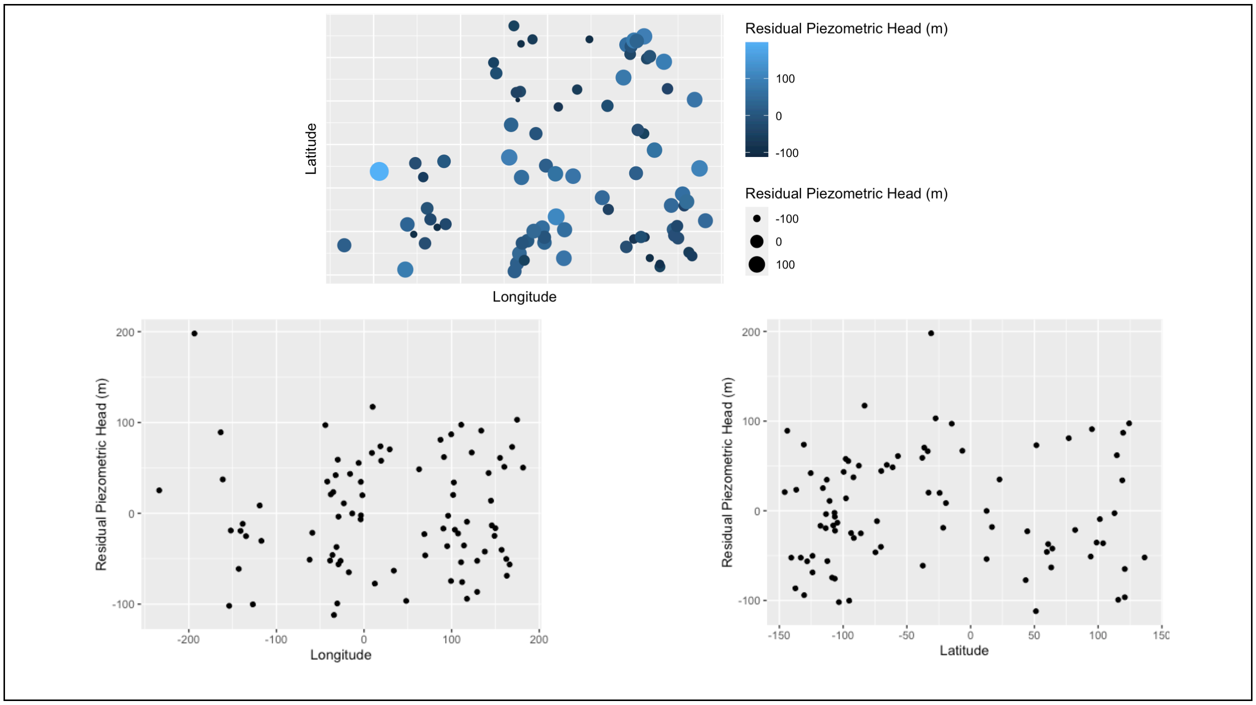



Figure 4: Visualization of the detrended data, including a bubble plot (top) and plot of the detrended data against longitude (bottom left) and latitude (bottom right).

Figure 4 confirms a lack of spatial dependence in the residuals; there are no trends apparent in the bubble plot of the residuals or scatter plot against longitude or latitude. Therefore, detrending was performed with a linear model of the form $z_s = \beta_0 + \beta_1 x_s + \beta_2 y_s + \varepsilon_s$, where $s$ is a specific observation with longitude $x_s$, latitude $y_s$, and piezometric head $z_s$, and $\varepsilon_s$ are the residuals for each observation which will be analyzed further.

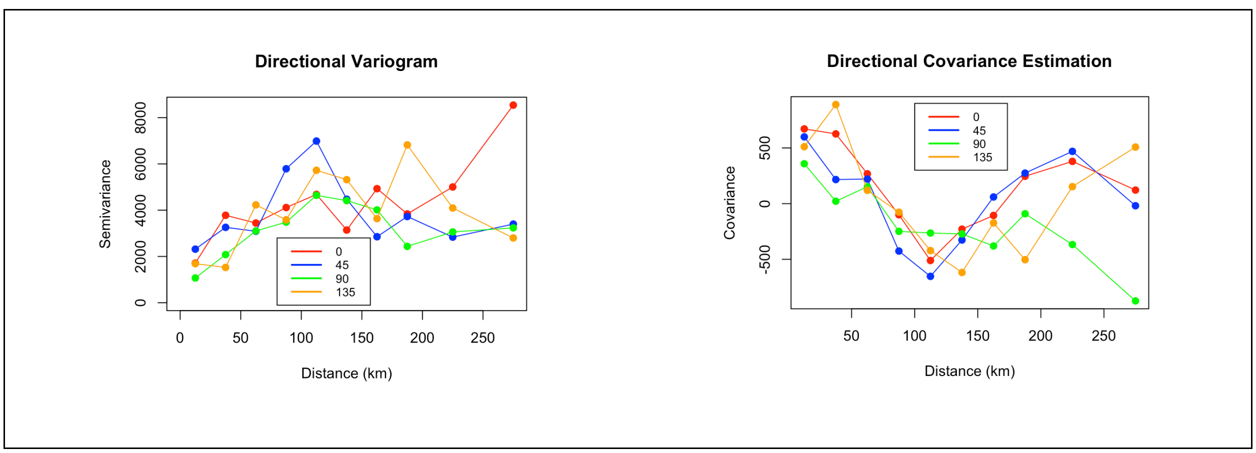



Figure 5: Plot of directional variogram (left) and directional covariance (right) for the detrended data.

The directional variogram and covariance for the detrended data are shown in Figure 5. A discussion of general features of the variogram and covariance is included below. Considering only directional dependence, no trends are readily apparent. From the directional variogram, observations at 90º appear to be slightly more correlated at short distances than in other directions. However, this effect is very slight, and disappears when decreasing the tolerance angle (see appendix, "Effect of decreasing tolerance angle"), suggesting that it may be due in large part to the cluster of observations in the central south that are angled somewhere between 90º and 135º relative to each other. Overall, the residuals do not appear to contain any obvious directional dependence.

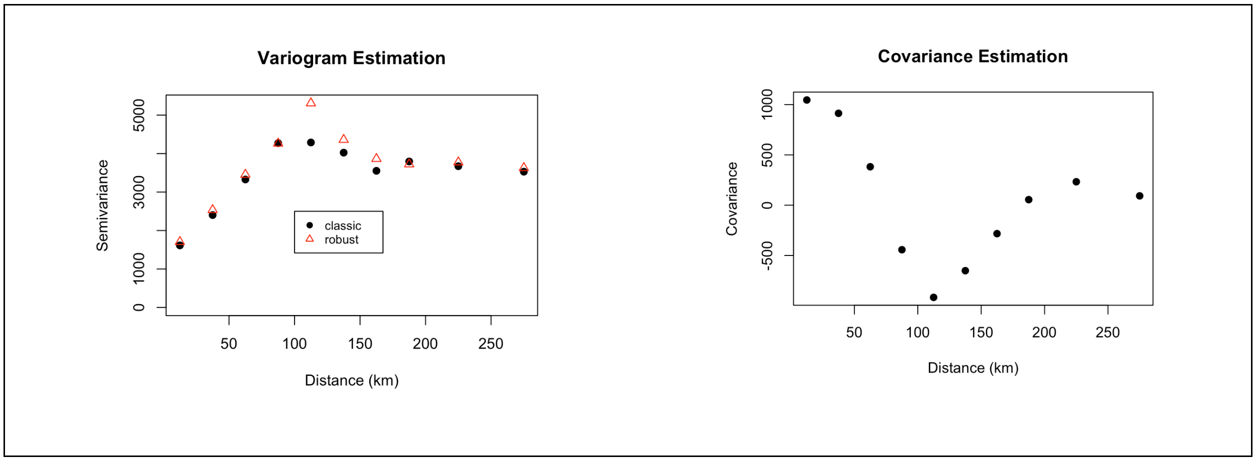



Figure 6: Plot of classic and robust empirical variogram (left) and empirical covariance (right) for the detrended data.

For fitting a variogram, a maximum distance of 300 km was used; past 300 km, values become quite noisy and numbers of available points become small. Bins were chosen to keep the variogram reasonably smooth while keeping the number of pairs of points in each bin reasonably consistent (see appendix, "Empirical variogram"). The empirical variogram fit to the residuals is shown in the left panel of Figure 6; the classic and robust variograms give quite different values for several points, reflecting non-Gaussianity in the data as discussed previously. It is likely that the distances at which these methods give differing values, around 100-150 km, corresponds to the distance between the two bimodal centers. In acknowledgment of the fact that the data is not exactly Gaussian, the robust variogram is used. The variogram follows the usual shape: it has a nugget of around 1,500 $m^2$, then increases before leveling off at around 4,000 $m^2$ for distances of 150 km or greater.

The empirical covariance, shown in the right panel of Figure 6, shows that observations are positively correlated at short distances, negatively correlated at intermediate distances, and nearly uncorrelated at large distances. The presence of positive correlation at short distances is expected, as nearby points are likely to be subject to the same patterns in pressure that the linear model was unable to capture. Uncorrelation at large distances is also expected. The negative correlation at intermediate distances however suggests the existence of one or more "pockets" of correlated points separated by a distance of about 100-150 km from another "pocket" of correlated points with residuals of the opposite sign. This observation is also consistent with the presence of a bimodal distribution.

To the empirical variogram, a linear, spherical, Matérn, and wave variogram (in an attempt to capture the "hump" in the semivariance) were each fit with non-weighted least squares, weighted least squares, and Cressie-style weights (Figure 7) using the optim minimization function. Of these variograms, the spherical variogram with Cressie-style weights was found to have the lowest sum of squared errors across the first three points (see appendix, "Variogram fitting"). With this variogram, the nlm optimization function was found to perform slightly better than optim, so the final variogram was chosen to be a spherical variogram of the form $\gamma(u) = \tau^2 + \sigma^2 \left[\frac{3}{2}\phi u - \frac{1}{2}(\phi u)^3\right]$.

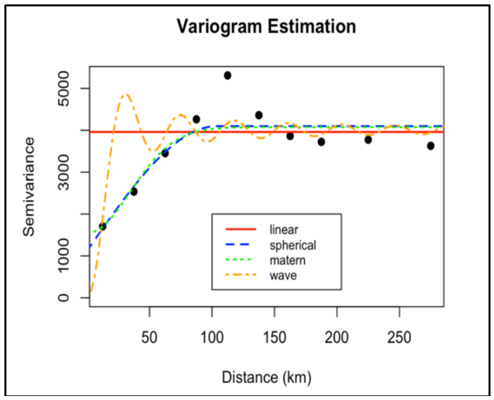



Figure 7: Plot of linear, spherical, Matérn, and wave variograms with Cressie-style weights fit to the robust empirical variogram. The spherical variogram was chosen as the best based on its fit to the first three points.

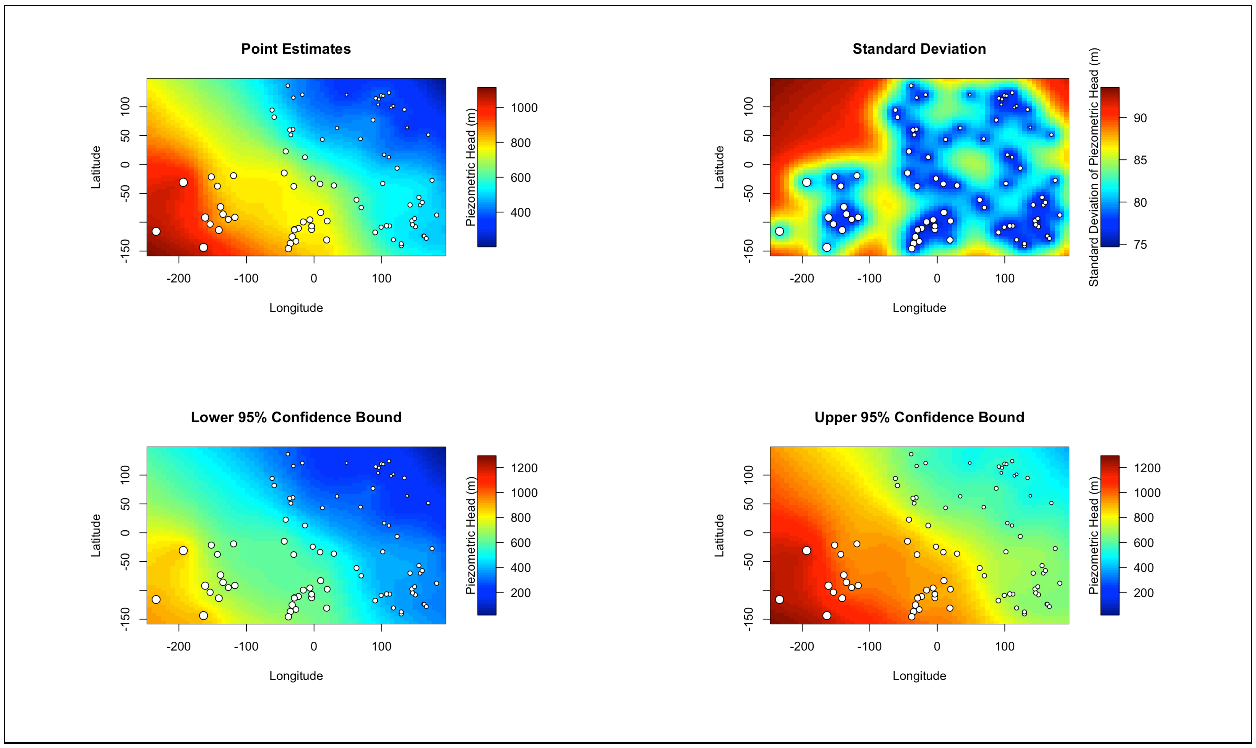



Figure 8: Results of kriging on the original scale, including point estimates (top left), standard deviations (top right), and lower and upper bounds for a 95% confidence interval (bottom left and right). The grid for prediction consists of 100 evenly spaced points spanning 10 km below the lowest $x$ or $y$ value to 10 km above the highest.

Using this variogram, kriging was performed on the residuals (see appendix, "Kriging," for residual point estimates, standard deviations, and confidence intervals) and was converted to the original scale by adding the point estimates and variances of the linear trend to the residual kriging estimates. Figure 8 shows point estimates, standard deviation, and lower and upper 95% confidence intervals for piezometric head values on the original scale. As one would expect, the standard deviation is smallest near points for which data was collected, and increases moving away from those points. The standard deviation does not increase dramatically moving away from those points – the scale ranges from about 75 m to 95 m. Thankfully, the points in which we are most interested, sites 15 and 59 (see Figure S2), are near several other points and have relatively low standard deviation, which should allow conclusions to be drawn about them more easily.

From the maps of point estimates and confidence intervals, several features are noticeable. First, the trend is the dominant feature; piezometric head values tend to be highest in the southwest and lowest in the northeast. Bands of equal pressure therefore run between the northwest and southeast, and kriging is able to capture several nonlinearities in these bands. Notably, it is able to capture a curved high-pressure region that includes the point that was a large outlier in the linear trend (site 78, at approximately (-200 km, -30 km)). There appears to be an additional nonlinear pressure feature around (100 km, -50 km).

The validity of these prediction intervals relies on several assumptions. Most notably is the assumption of Gaussianity in the data. The advantage of assuming Gaussianity is that it is possible to construct confidence intervals on the original scale. However, as we have seen, the piezometric head data are only very approximately Gaussian, and instead appear to follow a bimodal distribution. Standard transformations (such as the log transformation) work well for normalizing skewed data but do not significantly improve the normality of this bimodal data; a more complex transformation might be more successful. The same is true for the residuals of the linear model fit to capture the trend in piezometric head; to produce accurate confidence intervals, it is necessary to assume the residuals follow a Gaussian distribution, when in fact they appear slightly right-skewed and there is an outlier (Figure 3). Further, representing the pressure distribution of the aquifer as a static two-dimensional potentiometric surface is a simplification of the true hydrologic and geological processes that shape water movement in an aquifer. A statistical analysis such as this is unable to provide an explanation, for example, of the cause of the two nonlinearities in equipotential lines discussed above, or of some geological feature that might represent some sort of "split" in the aquifer that causes the data to be distributed bimodally, and such information is likely to be informative in assessing water movement patterns in the aquifer. Finally, any conclusions drawn rely on the quality of the data obtained: essentially, whether the contracting company quantified the uncertainty in their measurements well enough to report all useful data, exclude all erroneous data, and to report data representative of the true potentiometric surface.

Figure S2 shows the location of the potential nuclear waste site (site 15, lower left point in right panel) and the city of Amarillo, Texas (site 59, upper right point in right panel) in relation to all the points surveyed. Based on the sample measurements, the potential nuclear waste site is at higher pressure than Amarillo, so if nuclear waste were to leak out of the storage site there is concern that it would diffuse to a populated area. Based on the kriging results, the predicted difference in pressure between the two sites is 202.4 m with a standard deviation of 111.9 m; the p-value for testing whether this value is greater than 0 m is 0.035, indicating marginal significance. Further, this result is based on a one-tailed test (only a positive gradient is of interest, as only a positive gradient could carry water from the waste site to Amarillo); a two-tailed test would give a p-value of 0.07, outside of the standard range for significance. There is an argument to be made for conducting a two-tailed test, as *any* movement of nuclear waste away from the storage site is likely to come into contact with civilization in some way. However, given the nature of the problem, it is desirable to minimize the probability of a type II error (of concluding that there is no pressure differential and therefore no risk of nuclear contamination of a residential area when in fact there is) at the expense of an increased risk of type I error, so a large $\alpha$ value is desired. The conclusion is therefore that a pressure differential does exist between sites 15 and 59.

Based on this conclusion, the recommendation is that the potential site in Deaf Smith county, Texas is not suitable for nuclear waste storage. The potential for contamination of the groundwater of nearby Amarillo, Texas should leakage occur is too great. Historically, Congress ultimately came to the same conclusion and dropped the Deaf Smith county site from consideration due to concerns of aquifer contamination[3]. Of the potential sites under study, only one at Yucca Mountain, Nevada remained under consideration, and this site was eventually chosen for nuclear waste storage in 2002[4].

**References**


1. Harper, W.V. & Furr, J.M. (1986) *Geostatistical analysis of potentiometric data in the Wolfcamp Aquifer of the Palo Duro Basin, Texas*, vol. 587. , BMI/ONWI. Springfield, VA: Office of Nuclear Waste Isolation, Battelle Memorial Institute.
2. Blaettler, K.G. (2019) How to Calculate the Piezometric Head. Sciencing. URL https://sciencing.com/calculate-piezometric-head-8710823.html.
3. Nuclear Waste Management: An Inventory of the Records, 1972-1990, at the Southwest Collection/Special Collections Library (2020) Texas Archival Resources Online. URL https://legacy.lib.utexas.edu/taro/ttusw/00079/tsw-00079.html.
4. President Signs Yucca Mountain Bill (2002) The White House.

## Supplemental Figures

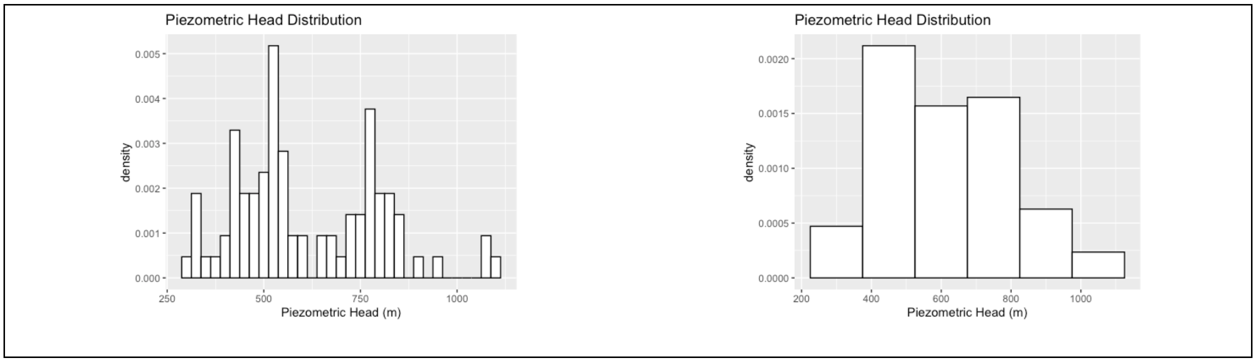



Figure S1: Exploratory data analysis: histograms of piezometric head distribution with different bin widths.

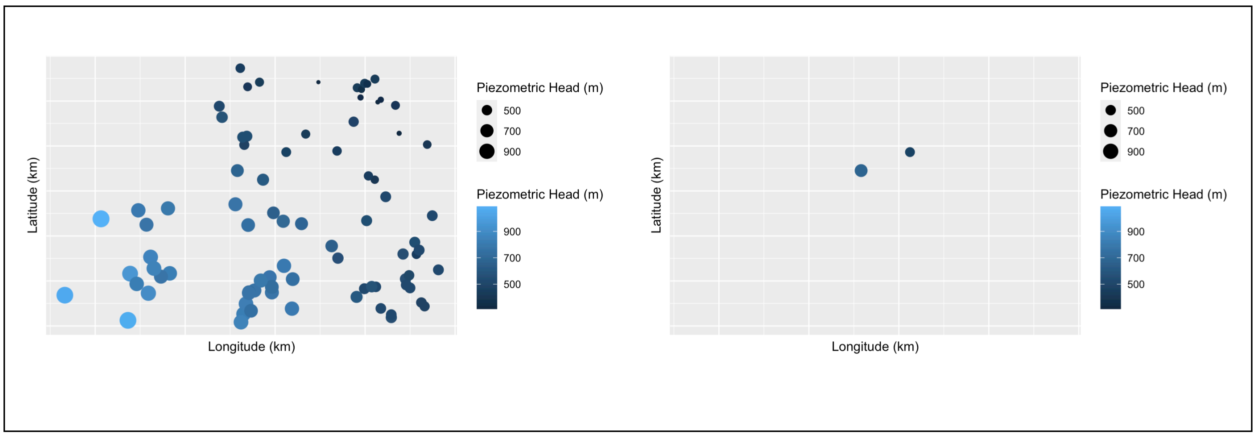
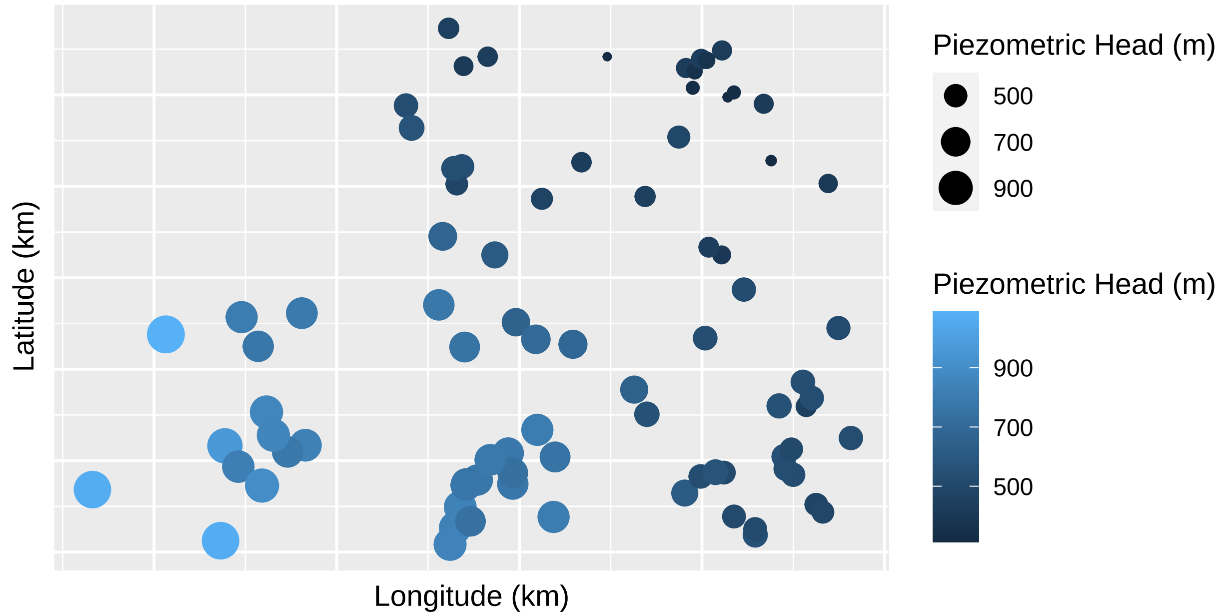


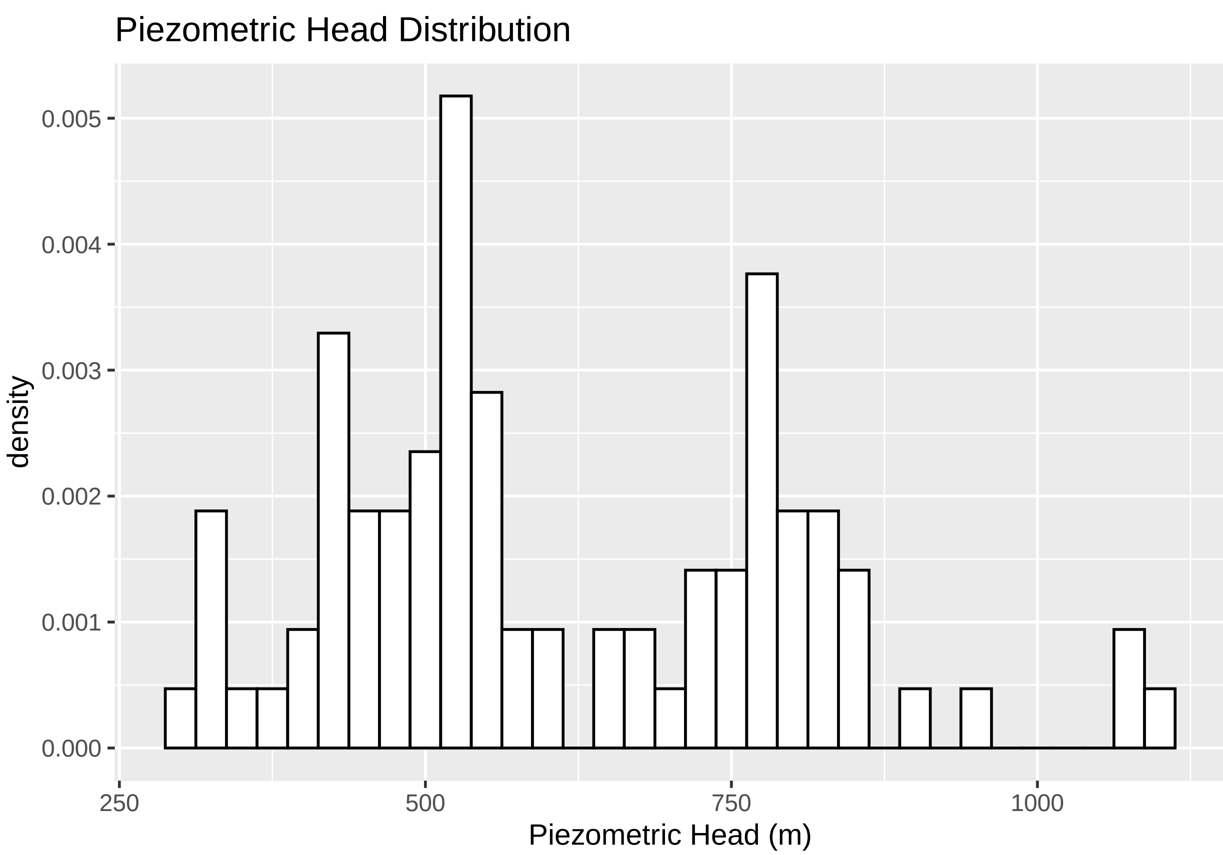

Figure S2: (Right panel) Location of the potential nuclear waste site (lower left point) and Amarillo, Texas (upper right point) relative to all locations surveyed (left panel).

# Appendix: All R Codes and Outputs

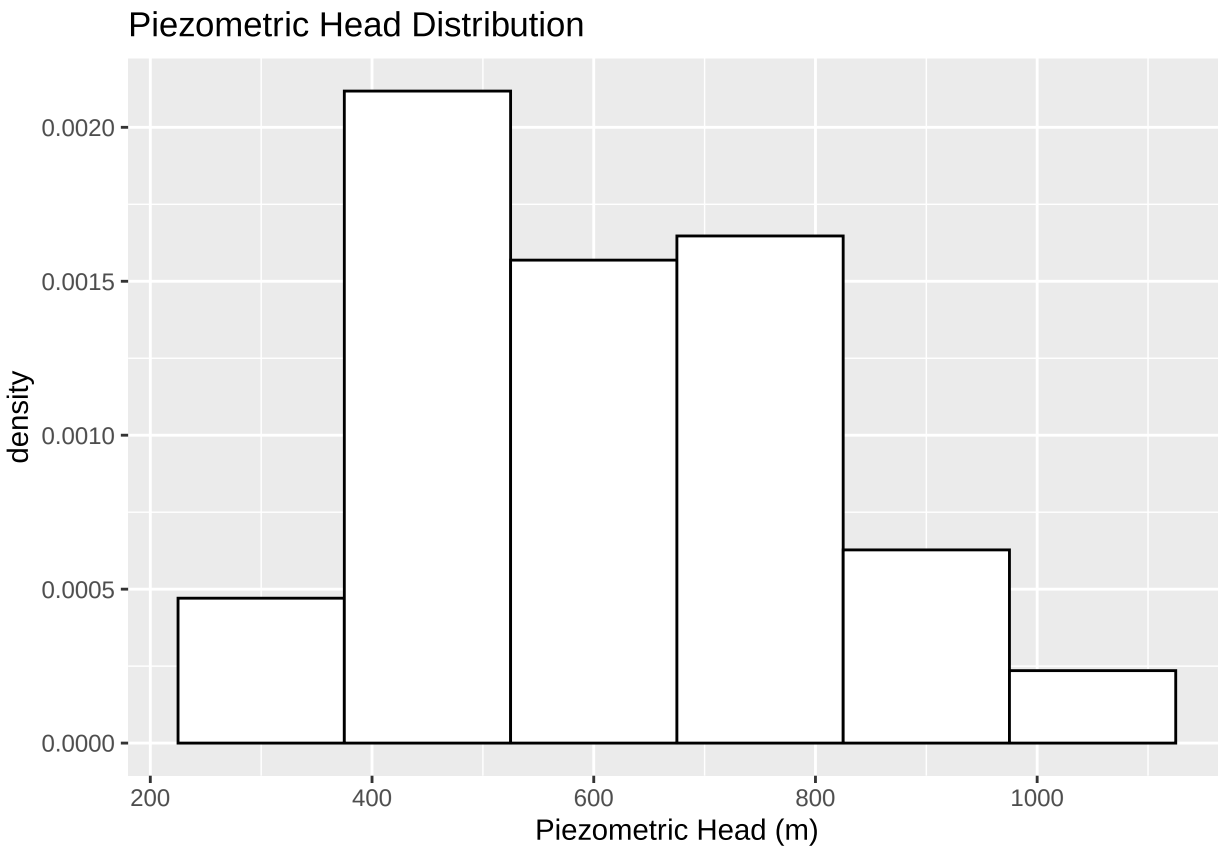
```
##################
# Data Exploration
##################

# Loading Data

library(geostatsp)
library(geoR)
library(fields)
library(ggplot2)

data(wolfcamp)

x = wolfcamp$coords[,1]
y = wolfcamp$coords[,2]
z = wolfcamp$data

# Setting up grid for later

grid = NULL
for (i in seq(min(x) - 10, max(x) + 10, length = 100))
  for (j in seq(min(y) - 10, max(y) + 10, length = 100))
    grid = rbind(grid, c(i, j))
colnames(grid) = c("LONGITUDE", "LATITUDE")

# Exploratory Analysis/Visualization

  # Bubble plot

df = data.frame(lat = y, long = x, Z = z)
p = ggplot() + geom_point(data = df, aes(x = long, y = lat, size = Z, color = Z))
p = p + labs(x = "Longitude (km)", y = "Latitude (km)", size = "Piezometric Head (m)",
  colour = "Piezometric Head (m)") + coord_fixed()
p = p + theme(axis.text.x = element_blank(), axis.text.y = element_blank(), axis.ticks =
  element_blank())
p
```



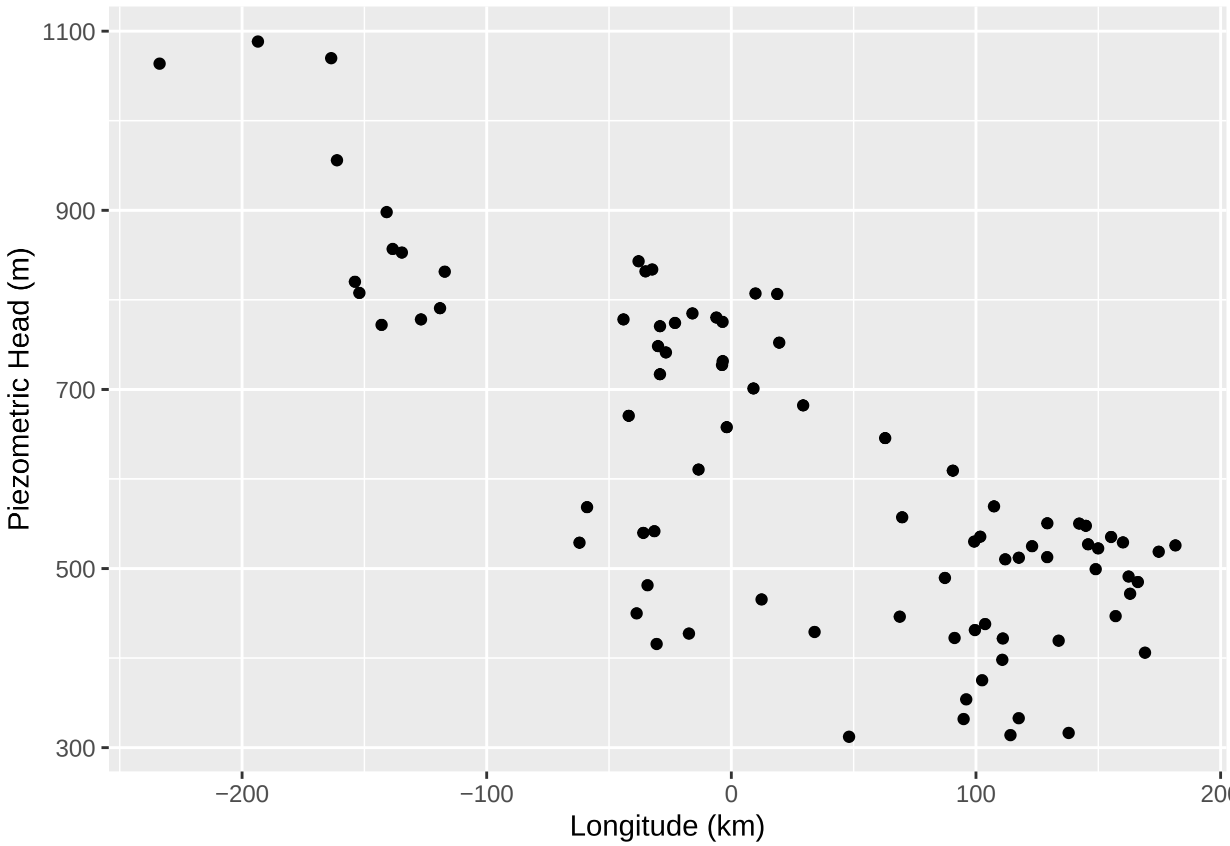

```
# Histograms

df = data.frame(X = z)
ggplot() + geom_histogram(data = df, aes(x = X, y = ..density..), binwidth = 25, color =
  "black", fill = "white") + ggtitle("Piezometric Head Distribution") +
  xlab("Piezometric Head (m)")
```



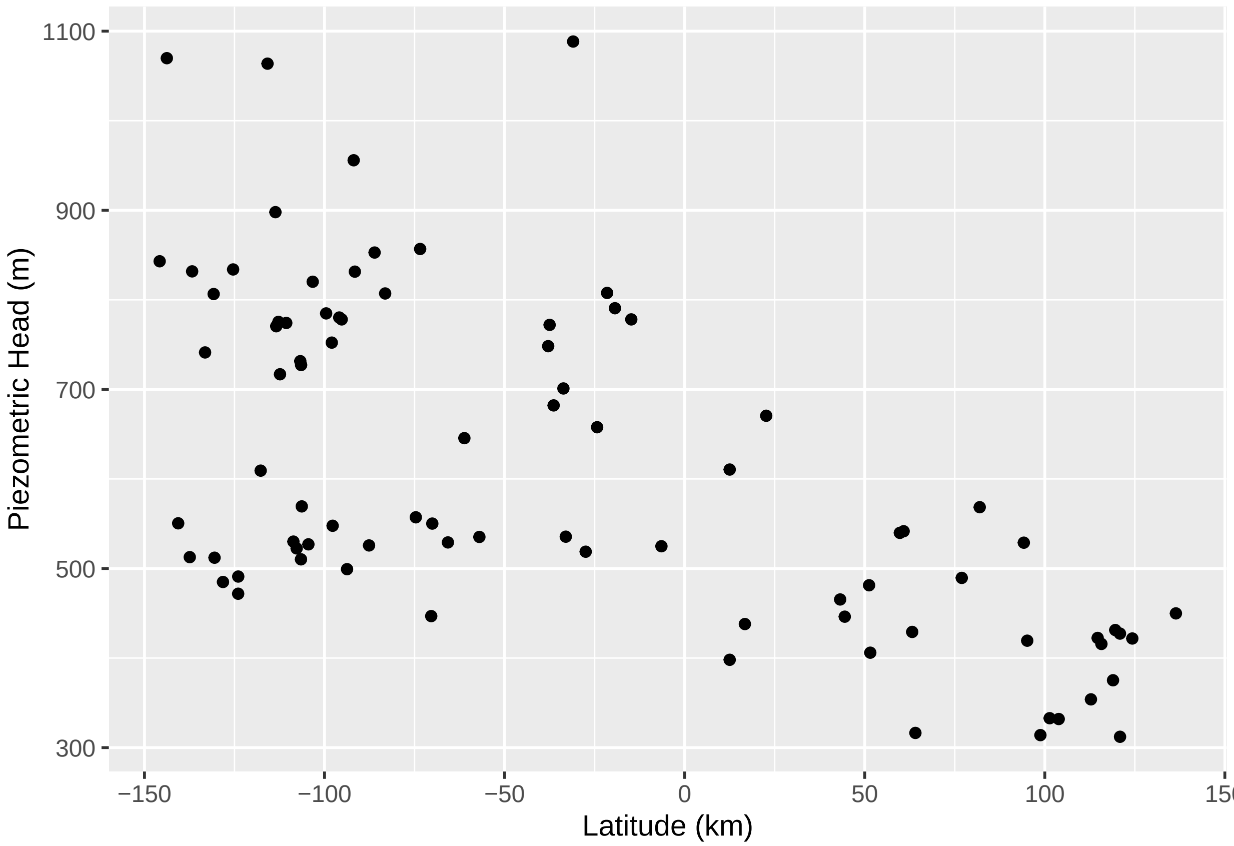
```r
ggplot() + geom_histogram(data = df, aes(x = X, y = ..density..), binwidth = 150, color =
  "black", fill = "white") + ggtitle("Piezometric Head Distribution") +
  xlab("Piezometric Head (m)")
```




```r
  # Plot against latitude and longitude

df = data.frame(X = x, Y = y, Z = z)
ggplot(df, aes(X, Z)) + geom_point() + ylab("Piezometric Head (m)") +
  xlab("Longitude (km)")
```




```
ggplot(df, aes(Y, Z)) + geom_point() + ylab("Piezometric Head (m)") +
  xlab("Latitude (km)")
```

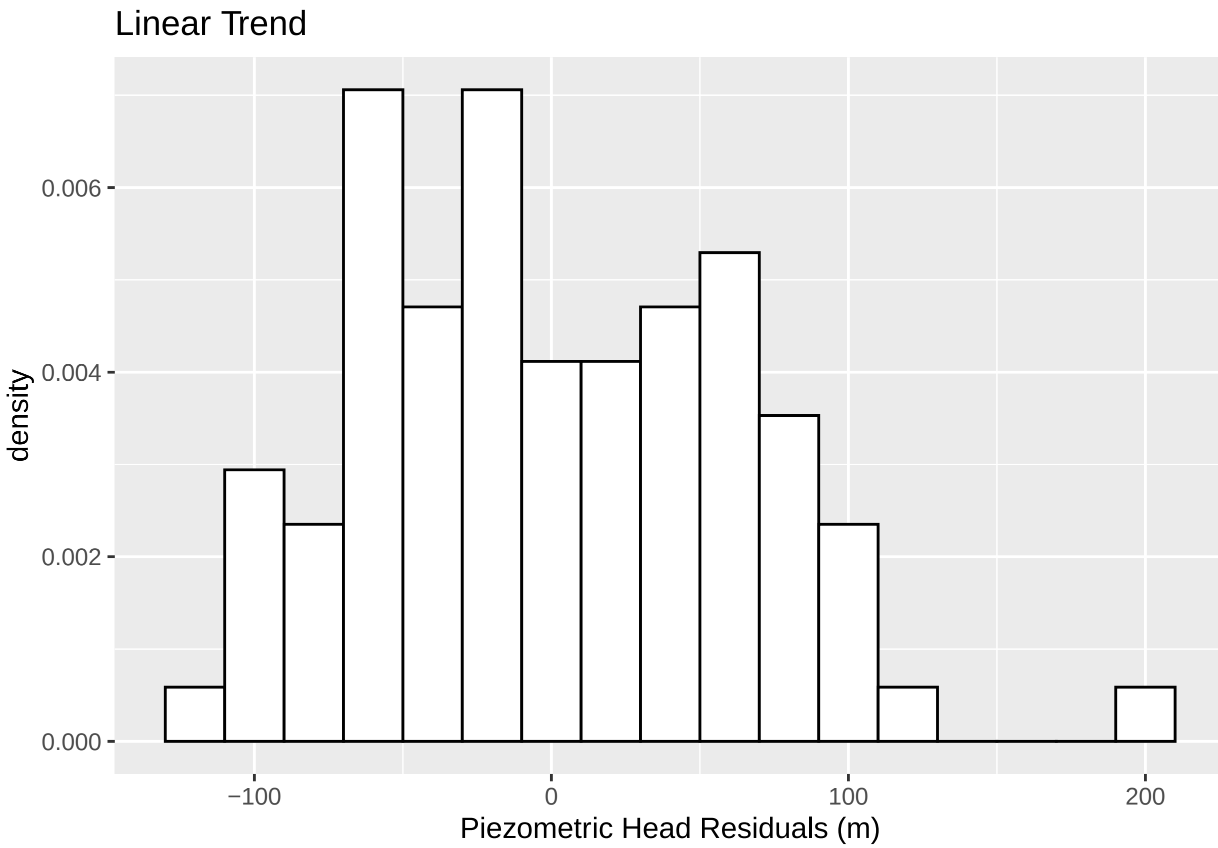



```
  # Fit a linear trend

mod.z = lm(z ~ x + y)
summ.z = summary(mod.z)
summ.z
```

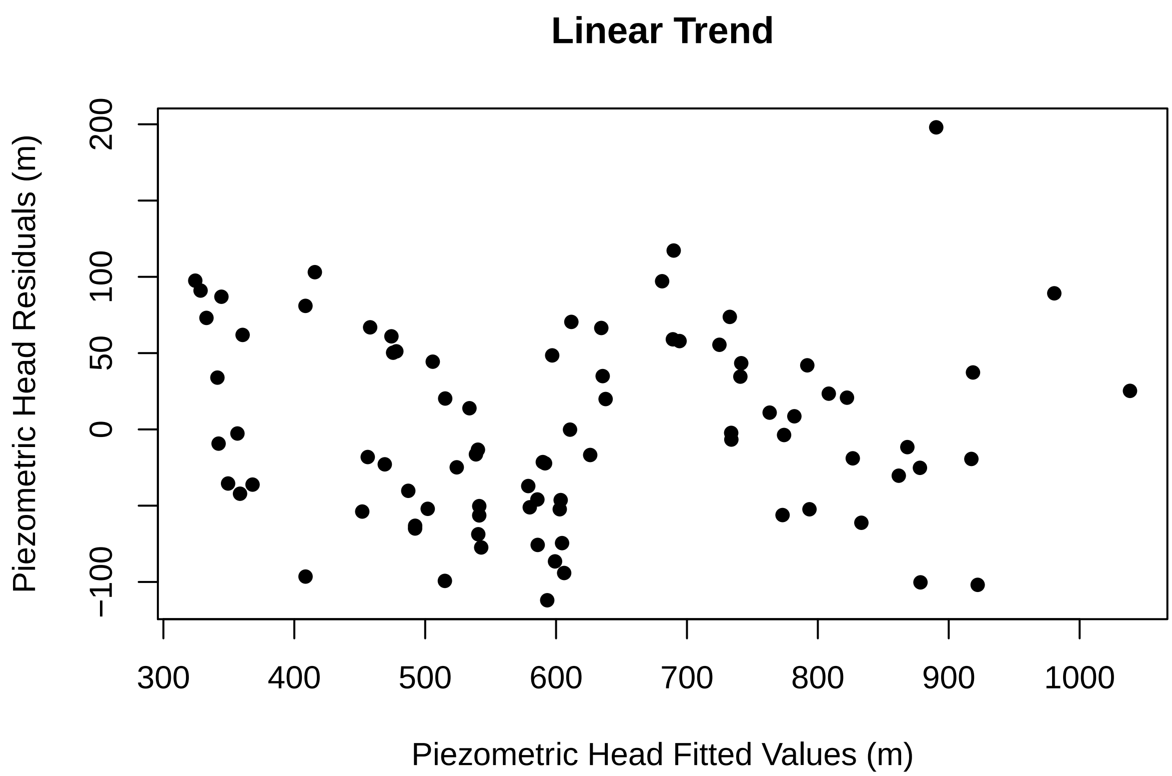
```
##
## Call:
## lm(formula = z ~ x + y)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -111.989  -50.297   -9.326   48.510  197.986
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 607.77066    7.52219   80.80   <2e-16 ***
## x            -1.27844    0.06552  -19.51   <2e-16 ***
## y            -1.13874    0.07739  -14.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 62.29 on 82 degrees of freedom
## Multiple R-squared:  0.8909, Adjusted R-squared:  0.8882
## F-statistic: 334.8 on 2 and 82 DF,  p-value: < 2.2e-16
```

```
ggplot() + geom_histogram(data = df, aes(x = summ.z$residuals, y = ..density..),
  binwidth = 20, color = "black", fill = "white") + ggtitle("Linear Trend") +
```

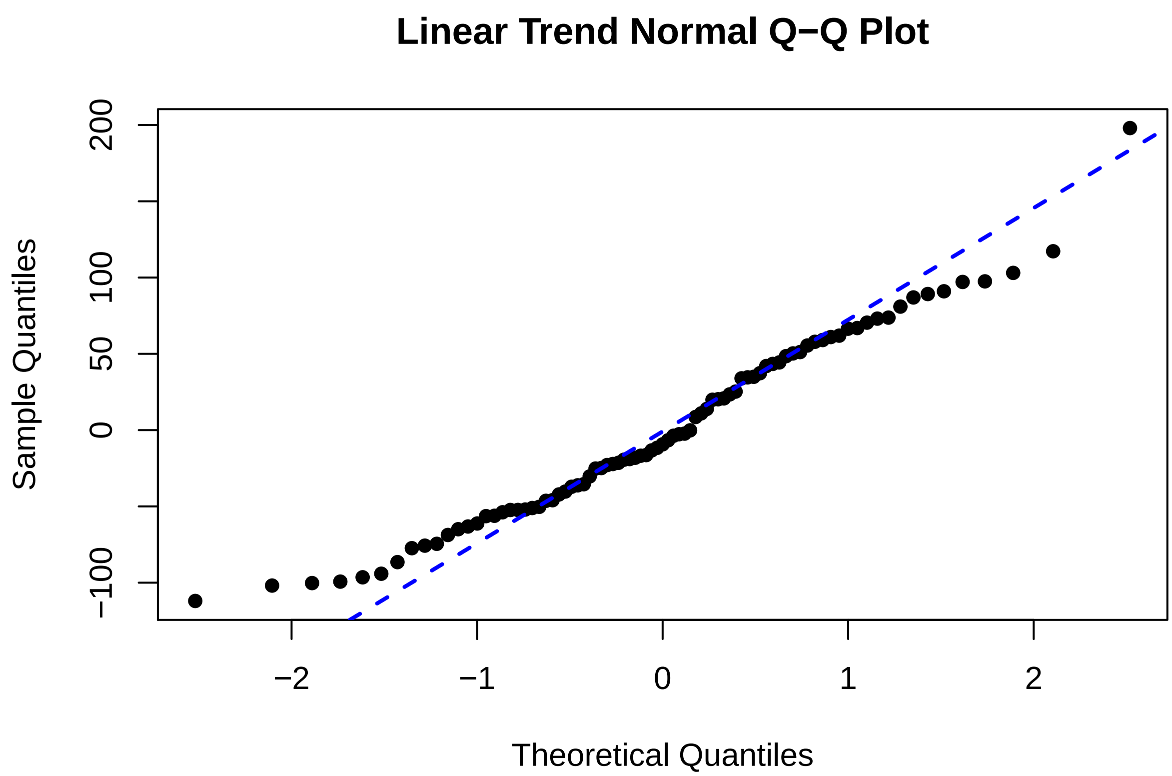
```
  xlab("Piezometric Head Residuals (m)")
```




```
plot(x = mod.z$fitted.values, y = summ.z$residuals, pch = 16, main = "Linear Trend",
  xlab = "Piezometric Head Fitted Values (m)", ylab = "Piezometric Head Residuals (m)")
```




```
qqnorm(summ.z$residuals, pch = 16, main = "Linear Trend Normal Q-Q Plot")
qqline(summ.z$residuals, lwd = 2, lty = 2, col = "blue")
```




```
# Linear trend without the outlier
```

```r
cbind(x, y, z, summ.z$residuals)
```

```
##              x           y        z
## 1    68.851186   44.453990 446.2190  -22.9078072
## 2   -44.090428  -14.826161 778.1401   97.1192701
## 3    -1.871464  -24.307187 657.7464   19.9035400
## 4   -29.962712  -37.896311 748.2703   59.0399579
## 5   155.243957  -57.001223 535.2190   61.0090994
## 6   174.711819  -27.481976 518.7601  103.0535097
## 7   142.205349  -70.080337 550.1539   44.3811979
## 8   144.906708  -97.753066 547.7156   13.8843026
## 9   149.940455 -107.744054 522.4176  -16.3554284
## 10  157.085841  -70.375616 446.8286  -40.2565094
## 11  145.836686 -104.477132 526.9895  -13.3097641
## 12  148.942676  -93.742959 499.2533  -24.8517851
## 13  160.095272  -65.740771 529.1231   51.1632536
## 14  -38.720268  136.406064 449.8766  -52.0645427
## 15  -41.938910   22.643149 670.5477   34.9452726
## 16   90.556503 -117.737938 609.2840  -16.7884035
## 17  117.528726 -130.548135 512.0546  -94.1228962
## 18  129.164119 -140.638579 550.4587  -52.3340263
## 19  129.115839 -137.419936 512.6642  -86.5250568
## 20  110.769577   12.504426 398.0615  -53.8573709
## 21  122.936046   -6.453378 524.8560   66.9032025
## 22  103.736844   16.720847 437.9896  -18.1188327
## 23   69.828446  -74.656410 557.1642  -46.3491649
## 24   62.876179  -61.170298 645.5546   48.5103336
## 25  181.531430  -87.643632 525.7704   50.2737146
## 26   87.305674   76.941646 489.4998   80.9610183
## 27    9.865139  -83.169719 807.0956  117.2281917
## 28   -6.115421  -95.915543 780.2737   55.4618712
## 29   -3.588786 -112.797322 775.3970   34.5913436
## 30   -3.797998 -106.504876 727.2395   -6.6681736
## 31   -3.508320 -106.730181 731.5066   -2.2872791
## 32  101.741286  -33.007178 535.5238   20.2368343
## 33  -17.332389  120.876114 427.3218  -64.9607576
## 34  -30.544916  115.726287 415.7396  -99.2987133
## 35  -62.071518   94.177476 528.8183  -51.0634321
## 36  133.798964   95.126975 419.3971   91.0056579
## 37  -35.083202 -136.760115 831.7840   23.4271114
## 38  -37.915607 -145.788406 843.0614   20.8025538
## 39  -32.363449 -125.382214 833.9175   41.9942013
## 40  -26.746918 -133.171328 741.2600  -52.3526760
## 41   48.118704  120.892208 312.1095  -96.4792933
## 42  162.396601 -123.966011 491.0238  -50.2974007
## 43  163.008143 -123.982104 471.8218  -68.7359535
## 44  166.178506 -128.198526 484.9279  -56.3780809
## 45  -23.029386 -110.624738 774.1778   10.9924965
## 46  -29.176993 -112.362805 716.8765  -56.1474207
## 47  -29.160900 -113.408864 770.5203   -3.6742200
## 48  -15.900093  -99.552609 784.8456   43.3829763
## 49   19.569346  -97.991567 752.2326   57.8932165
## 50   18.748592 -130.789533 806.4860   73.7489475
## 51   99.279024 -108.645273 530.0375  -74.5295238
```

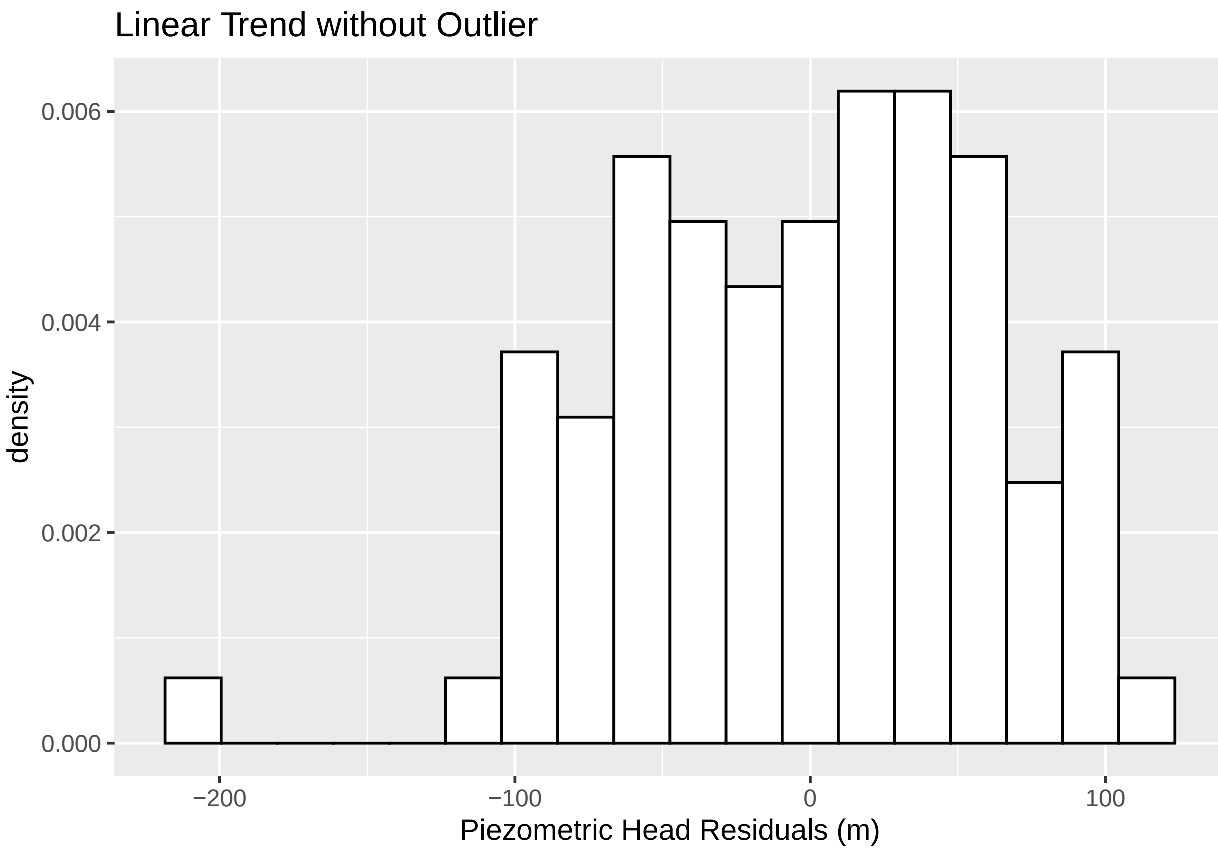
```
## 52  111.960475 -106.537063  510.2259  -75.7279560
## 53  107.373910 -106.327851  569.3560  -22.2232591
## 54  -58.981622   81.930542  568.4416  -21.4359911
## 55  -31.462229   60.800154  541.6197  -37.1379904
## 56  -34.262448   51.208600  481.2704 -111.9894987
## 57  -35.984422   59.754096  539.7909  -45.9393080
## 58   34.021050   63.181950  429.1505  -63.1783072
## 59   12.359587   43.178087  465.4211  -77.3799153
## 60  -13.405645   12.504426  610.5032   -0.1664825
## 61   91.248511  114.680228  422.4451   61.9214063
## 62   94.949950  103.865589  331.9211  -36.1855334
## 63  110.978789  124.303969  421.8355   97.4947819
## 64  114.100872   98.780135  313.9382  -35.4760753
## 65  117.480447  101.355048  332.8355   -9.3260726
## 66   96.012102  112.813415  353.8663   -2.6931794
## 67   99.568702  119.572564  431.2841   86.9684296
## 68  102.513760  118.944929  375.2019   33.9366287
## 69    9.044385  -33.666999  701.0272   66.4812252
## 70   29.354018  -36.386752  682.1299   70.4515648
## 71  137.886639   64.067077  316.3766  -42.1581790
## 72  169.091377   51.546558  405.9862   73.0871984
## 73 -163.571406 -143.792848 1069.8284   89.1983720
## 74 -233.721716 -115.838939 1063.7325   25.2516424
## 75 -119.073675  -19.360134  790.6367    8.5910886
## 76 -152.048666  -21.532717  807.7052  -18.9710474
## 77 -142.972094  -37.497184  772.0443  -61.2075166
## 78 -193.520873  -30.963340 1088.4209  197.9857612
## 79 -138.433809  -73.449419  856.7771  -11.6130184
## 80 -117.142489  -91.586469  831.4792  -30.3446401
## 81 -161.205703  -91.908333  955.8353   37.3126986
## 82 -126.862789  -95.239628  778.1401 -100.2705048
## 83 -134.684090  -86.098684  852.8148  -25.1857848
## 84 -153.867199 -103.286234  820.2018 -101.8954470
## 85 -140.896070 -113.634169  897.9244  -19.3736527
```

```
mod.z.dim = lm(z[-78] ~ x[-78] + y[-78])
summ.z.dim = summary(mod.z.dim)
summ.z.dim
```

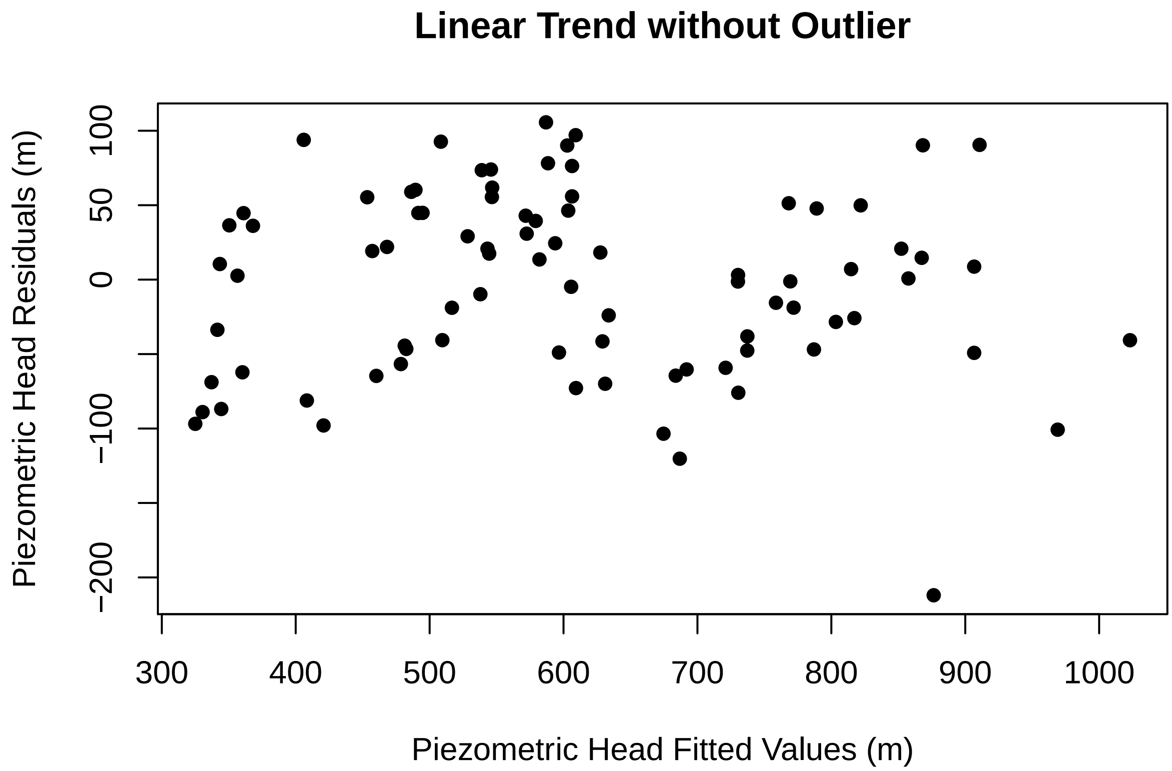
```
##
## Call:
## lm(formula = z[-78] ~ x[-78] + y[-78])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -105.641  -45.220   -5.076   47.089  120.267
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 603.58977    7.15170   84.40   <2e-16 ***
## x[-78]       -1.22650    0.06319  -19.41   <2e-16 ***
## y[-78]       -1.14632    0.07258  -15.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 58.4 on 81 degrees of freedom
## Multiple R-squared:  0.8971, Adjusted R-squared:  0.8946
## F-statistic: 353.2 on 2 and 81 DF,  p-value: < 2.2e-16
```

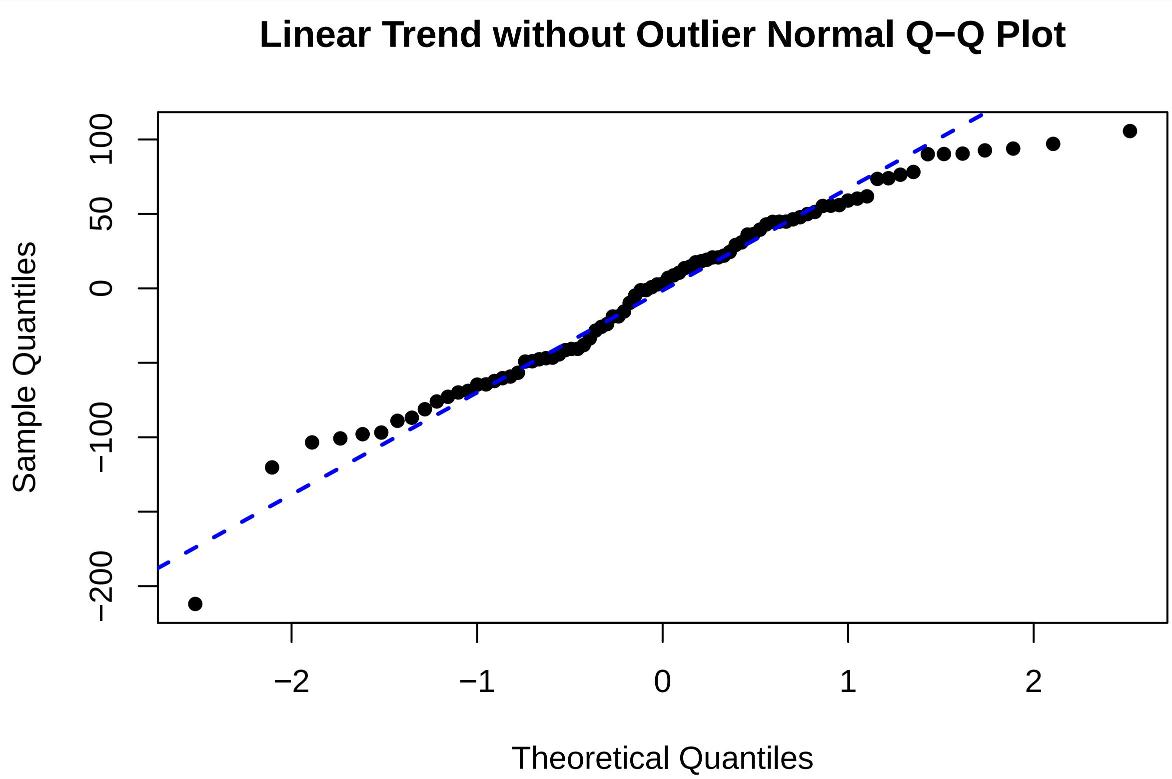
```
pred = cbind(1, x, y) %*% mod.z.dim$coefficients
resid = pred - z

ggplot() + geom_histogram(data = df, aes(x = resid, y = ..density..), binwidth = 19,
  color = "black", fill = "white") + ggtitle("Linear Trend without Outlier") +
  xlab("Piezometric Head Residuals (m)")
```




```
plot(x = pred, y = resid, pch = 16, main = "Linear Trend without Outlier", xlab =
  "Piezometric Head Fitted Values (m)", ylab = "Piezometric Head Residuals (m)")
```




```r
qqnorm(resid, pch = 16, main = "Linear Trend without Outlier Normal Q-Q Plot")
qqline(resid, lwd = 2, lty = 2, col = "blue")
```

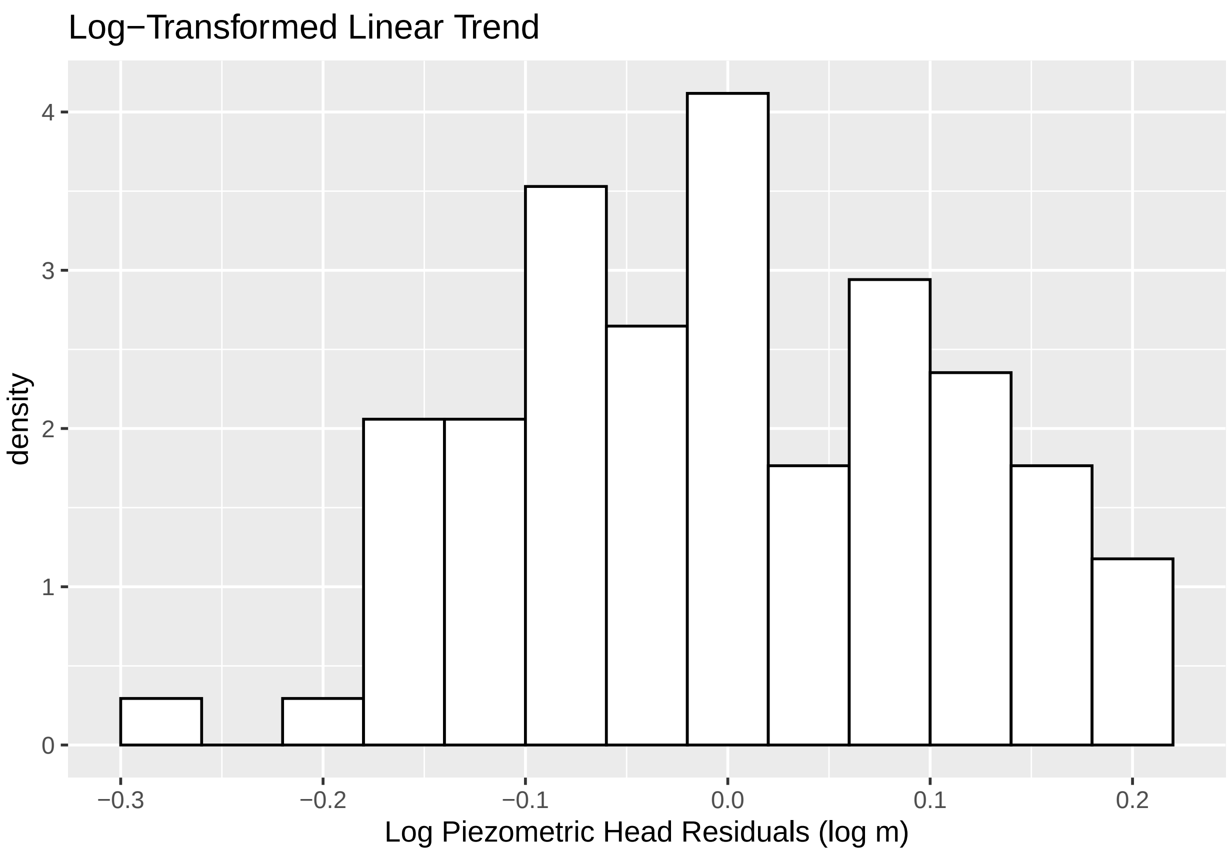



```r
# Log-transformed linear model
```

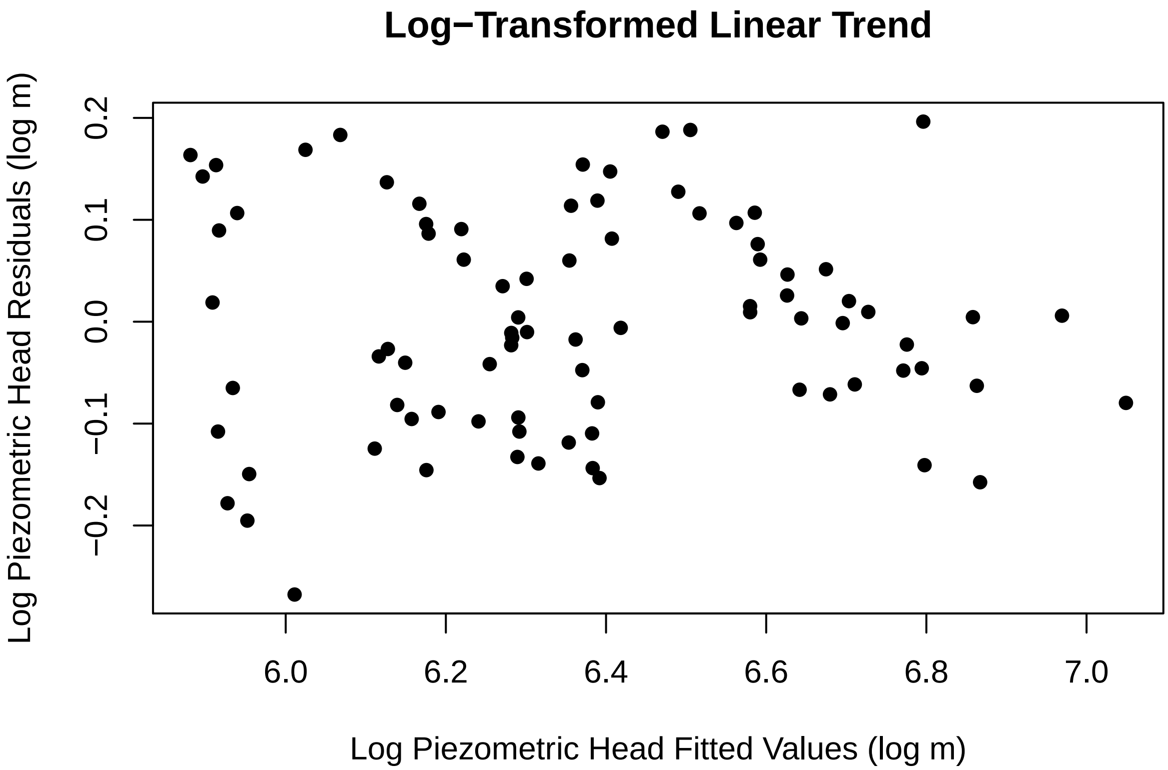
```r
mod.z.log = lm(log(z) ~ x + y)
summ.z.log = summary(mod.z.log)
summ.z.log
```

```
##
## Call:
## lm(formula = log(z) ~ x + y)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.267750 -0.079684 -0.001379  0.089537  0.196348
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.3536945  0.0129129  492.04   <2e-16 ***
## x           -0.0019575  0.0001125  -17.40   <2e-16 ***
## y           -0.0020547  0.0001328  -15.47   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1069 on 82 degrees of freedom
## Multiple R-squared:  0.8815, Adjusted R-squared:  0.8786
## F-statistic: 304.9 on 2 and 82 DF,  p-value: < 2.2e-16
```

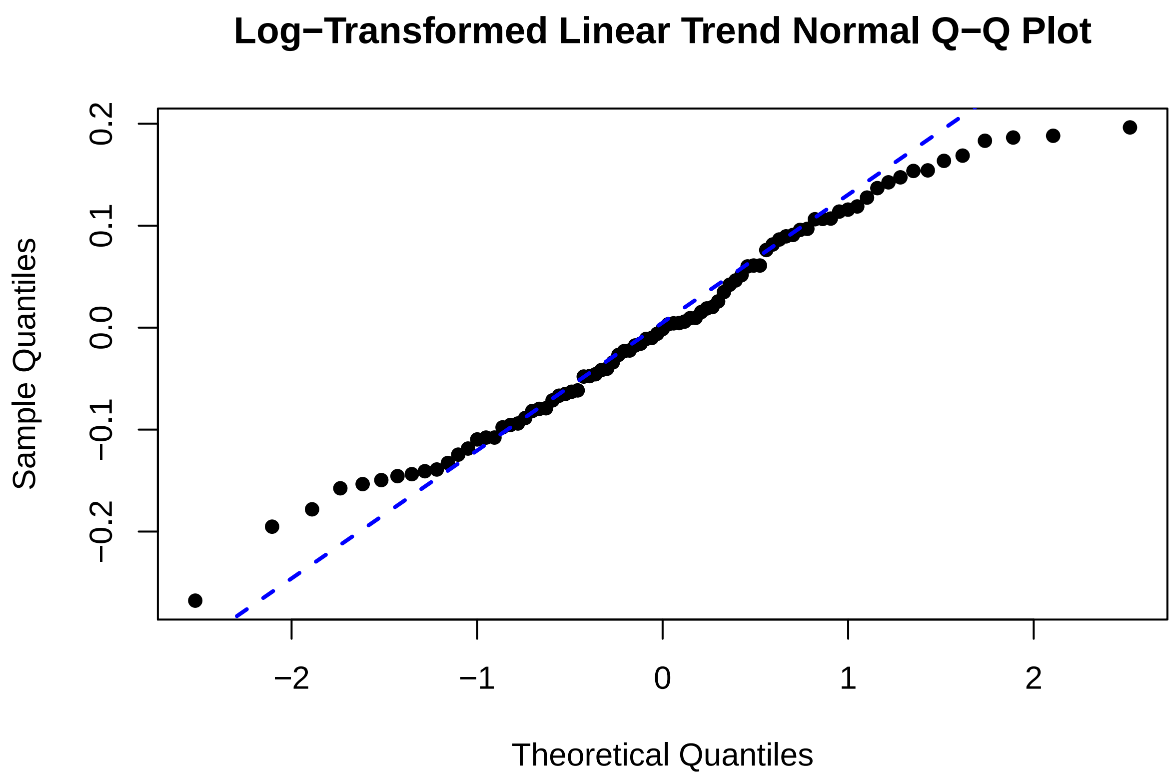
```r
ggplot() + geom_histogram(data = df, aes(x = summ.z.log$residuals, y = ..density..),
  binwidth = 0.04, color = "black", fill = "white") +
  ggtitle("Log-Transformed Linear Trend") + xlab("Log Piezometric Head Residuals (log m)")
```



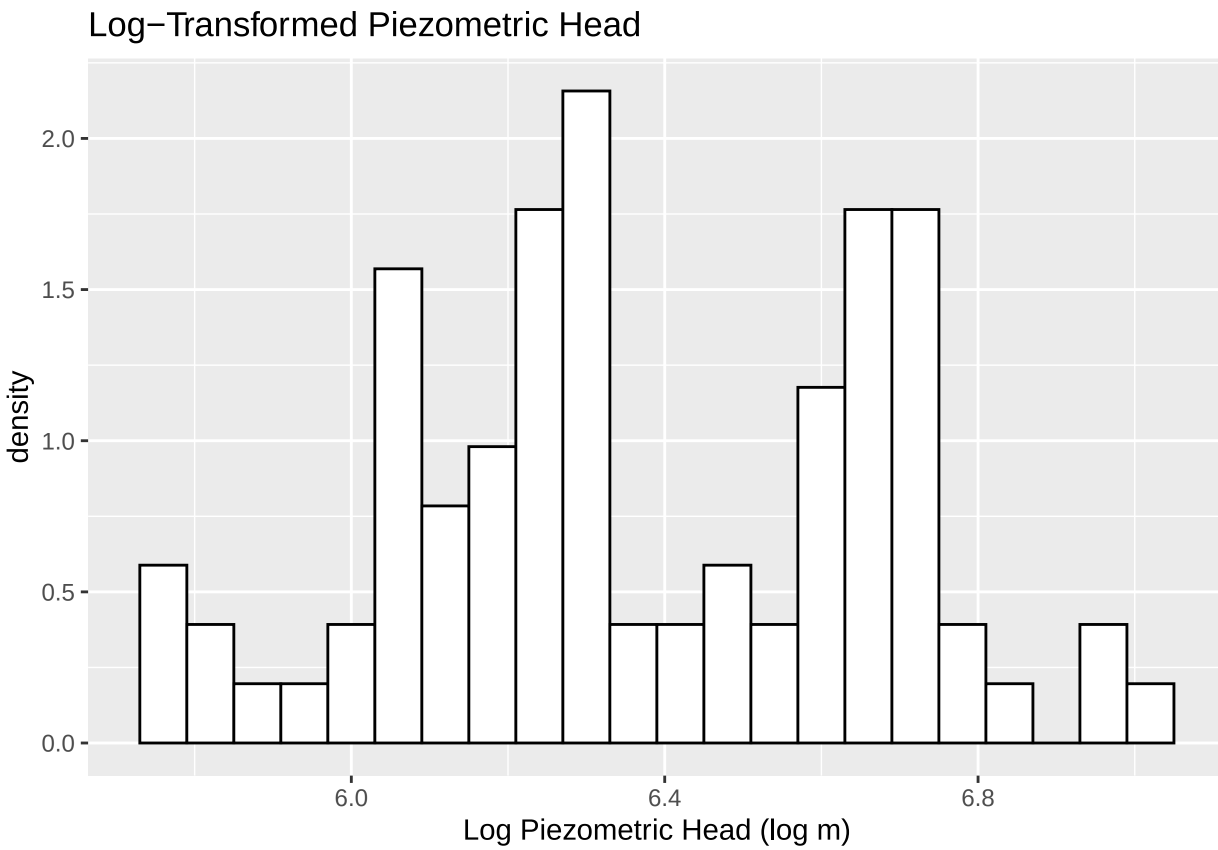
```r
plot(x = mod.z.log$fitted.values, y = summ.z.log$residuals, pch = 16, main =
  "Log-Transformed Linear Trend", xlab = "Log Piezometric Head Fitted Values (log m)",
  ylab = "Log Piezometric Head Residuals (log m)")
```



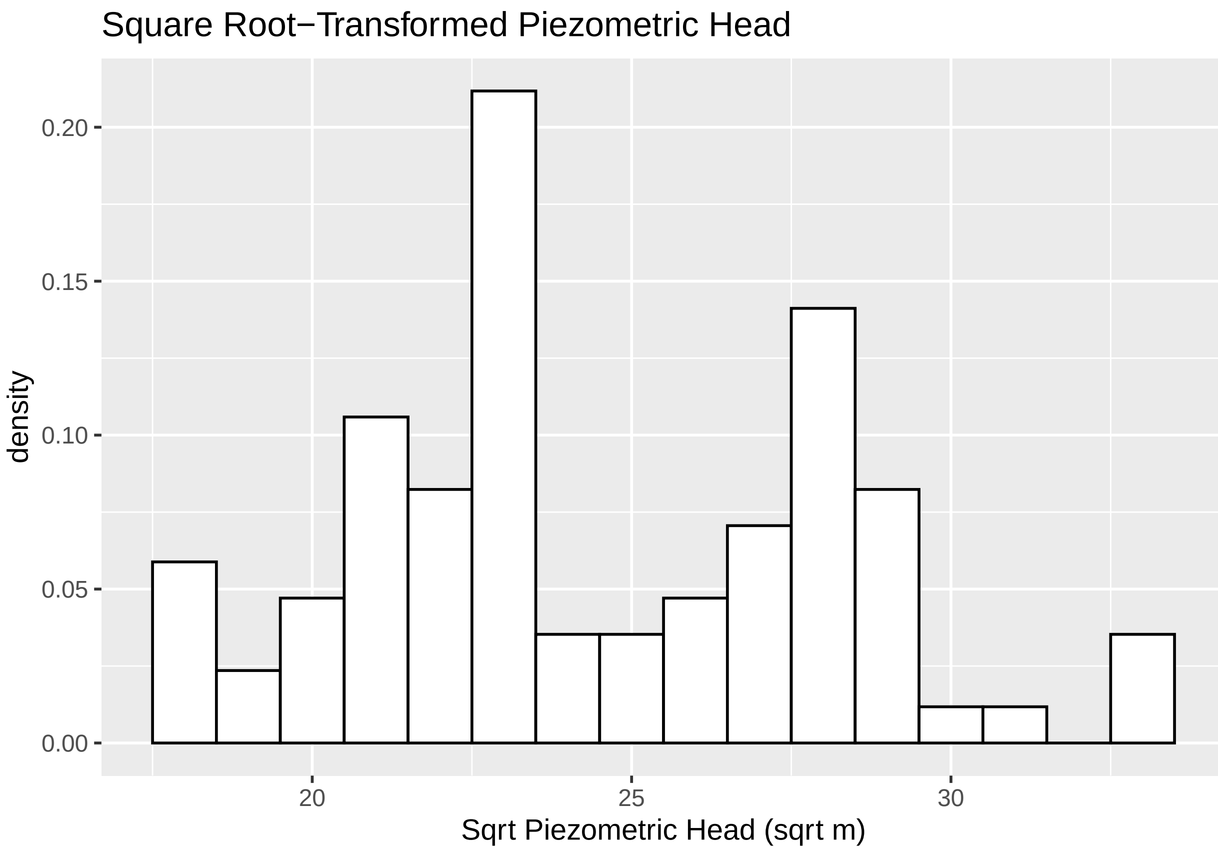

```r
qqnorm(summ.z.log$residuals, pch = 16, main =
  "Log-Transformed Linear Trend Normal Q-Q Plot")
qqline(summ.z.log$residuals, lwd = 2, lty = 2, col = "blue")
```



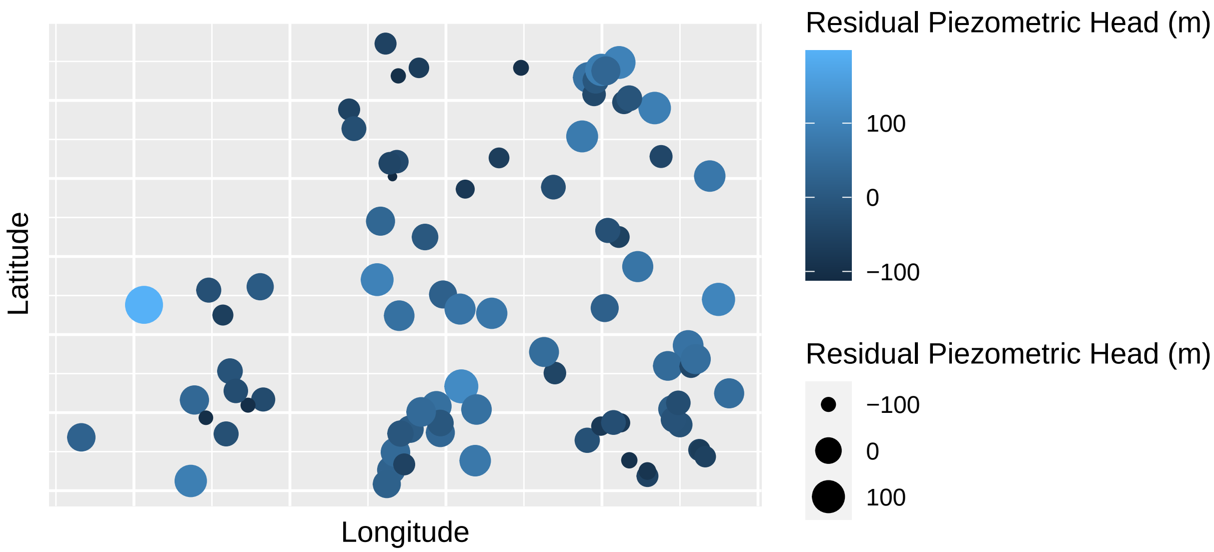

```
  # Log transform

ggplot() + geom_histogram(data = df, aes(x = log(z), y = ..density..), binwidth = 0.06,
  color = "black", fill = "white") + ggtitle("Log-Transformed Piezometric Head") +
  xlab("Log Piezometric Head (log m)")
```



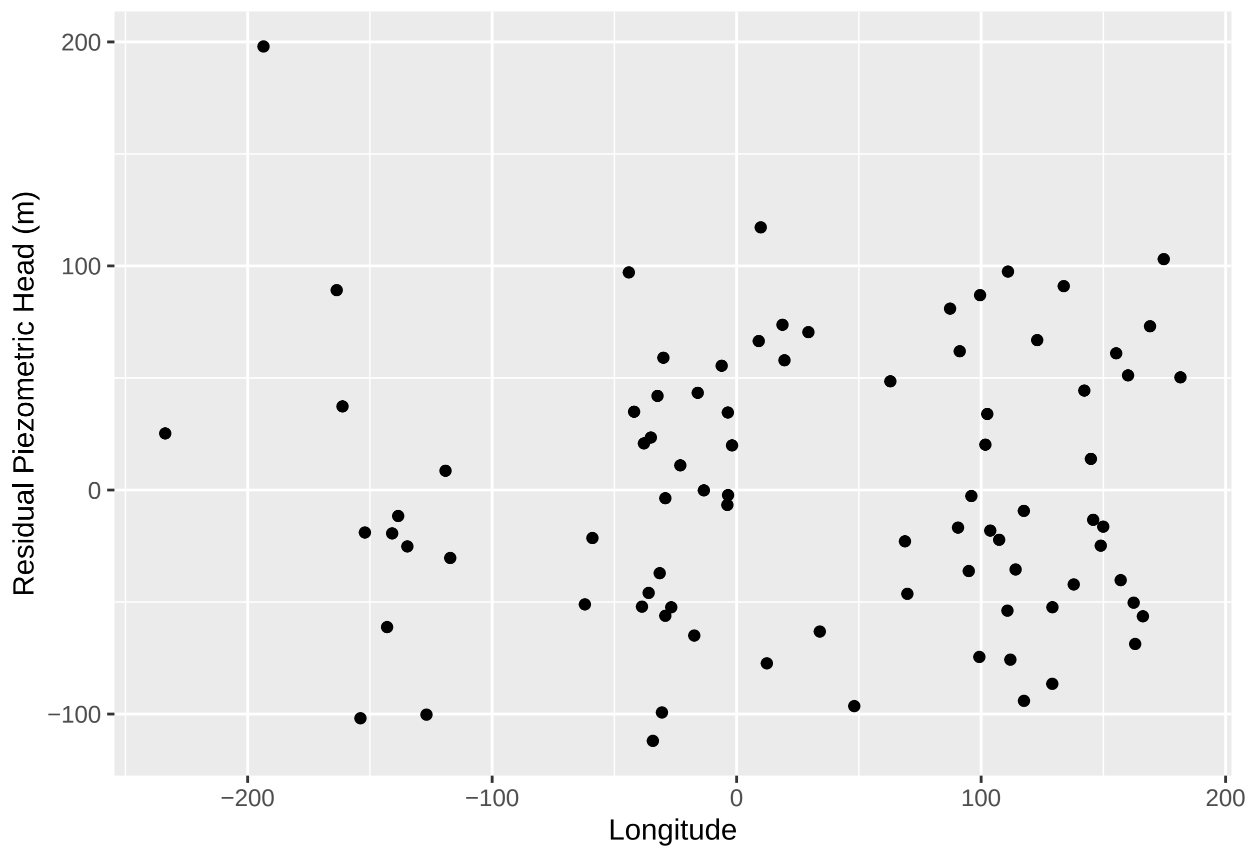

```r
# Square root transform

ggplot() + geom_histogram(data = df, aes(x = sqrt(z), y = ..density..), binwidth = 1,
  color = "black", fill = "white") + ggtitle("Square Root-Transformed Piezometric Head") +
  xlab("Sqrt Piezometric Head (sqrt m)")
```



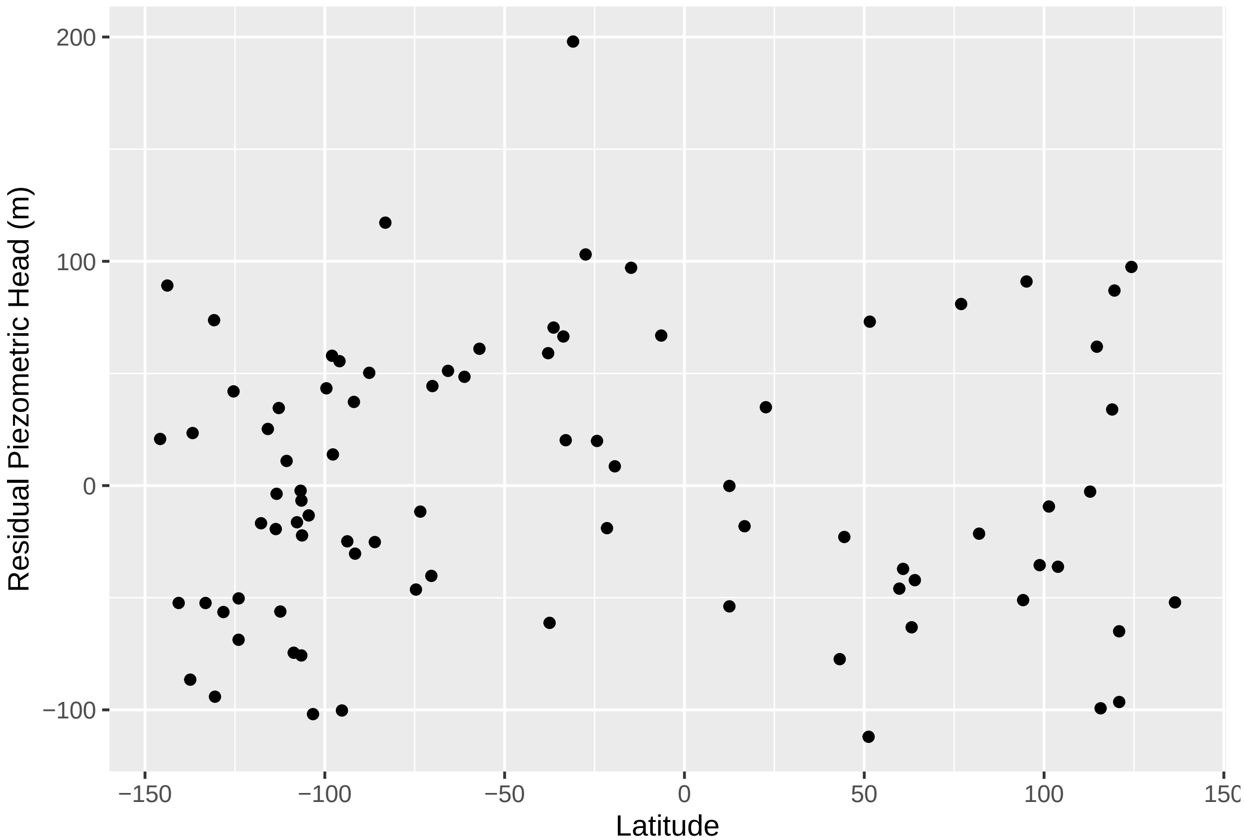

```
##################
# Residual Analysis
##################

  # Spatial plot of the residuals

z.trend = mod.z$fitted.values
z.resid = summ.z$residuals

df = data.frame(lat = y, long = x, Z = z.resid)
p = ggplot() + geom_point(data = df, aes(x = long, y = lat, size = Z, color = Z))
p = p + labs(x = "Longitude", y = "Latitude", size = "Residual Piezometric Head (m)",
  colour = "Residual Piezometric Head (m)") + coord_fixed()
p = p + theme(axis.text.x = element_blank(), axis.text.y = element_blank(), axis.ticks =
  element_blank())
p
```




```
# Plot the residuals against latitude and longitude

df = data.frame(X = x, Y = y, Z = z.resid)
ggplot(df, aes(X, Z)) + geom_point() + ylab("Residual Piezometric Head (m)") +
  xlab("Longitude")
```



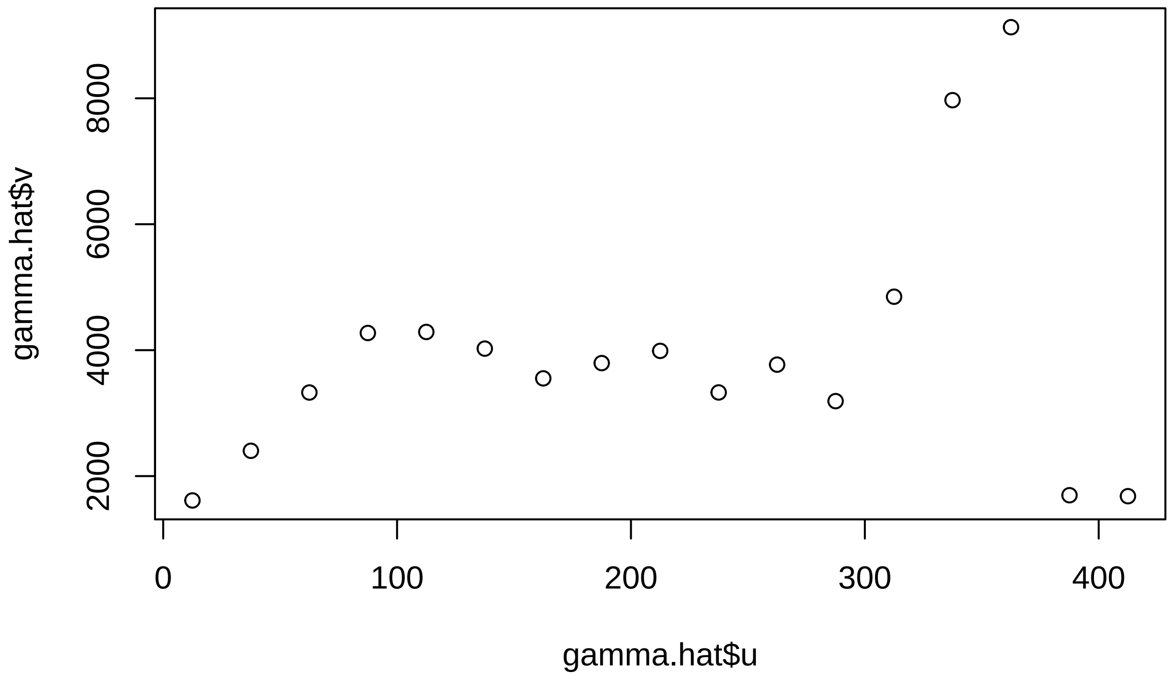

```
ggplot(df, aes(Y, Z)) + geom_point() + ylab("Residual Piezometric Head (m)") +
  xlab("Latitude")
```




```
  # Storing the data

wolfcamp.orig = wolfcamp
wolfcamp$data = z.resid

  # Prediction intervals for linear trend

pred = predict(mod.z, newdata = data.frame(x = grid[,1], y = grid[,2]), interval =
  "prediction", level = 0.95)
pred.point = pred[,1]
pred.sd = (pred[,3] - pred[,1]) / qnorm(0.975)


##################################
# Empirical Variogram & Covariance
##################################

  # Empirical Variogram

breaks = seq(0, 500, 25)
gamma.hat = variog(wolfcamp, breaks = breaks, estimator.type = "classical")
```

```
## variog: computing omnidirectional variogram
```

```
plot(x = gamma.hat$u, y = gamma.hat$v)
```



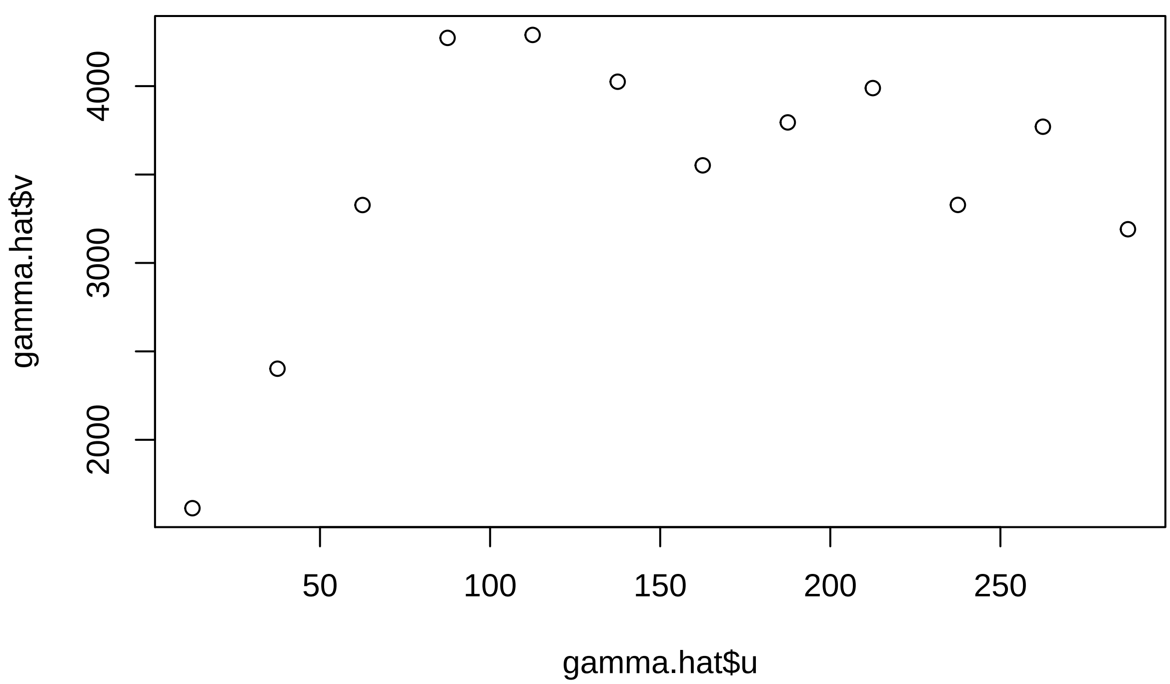

```r
gamma.hat$n
```

```
##  [1] 110 213 205 243 292 347 451 391 307 282 262 186 148  76  29  16  11
```

```r
breaks = seq(0, 300, 25)
gamma.hat = variog(wolfcamp, breaks = breaks, estimator.type = "classical")
```

```
## variog: computing omnidirectional variogram
```

```r
plot(x = gamma.hat$u, y = gamma.hat$v)
```




```r
gamma.hat$n
```

```
##  [1] 110 213 205 243 292 347 451 391 307 282 262 186
```

```r
breaks = c(seq(0, 200, 25), 250, 300)

gamma.hat = variog(wolfcamp, breaks = breaks, estimator.type = "classical")
```

```
## variog: computing omnidirectional variogram
```

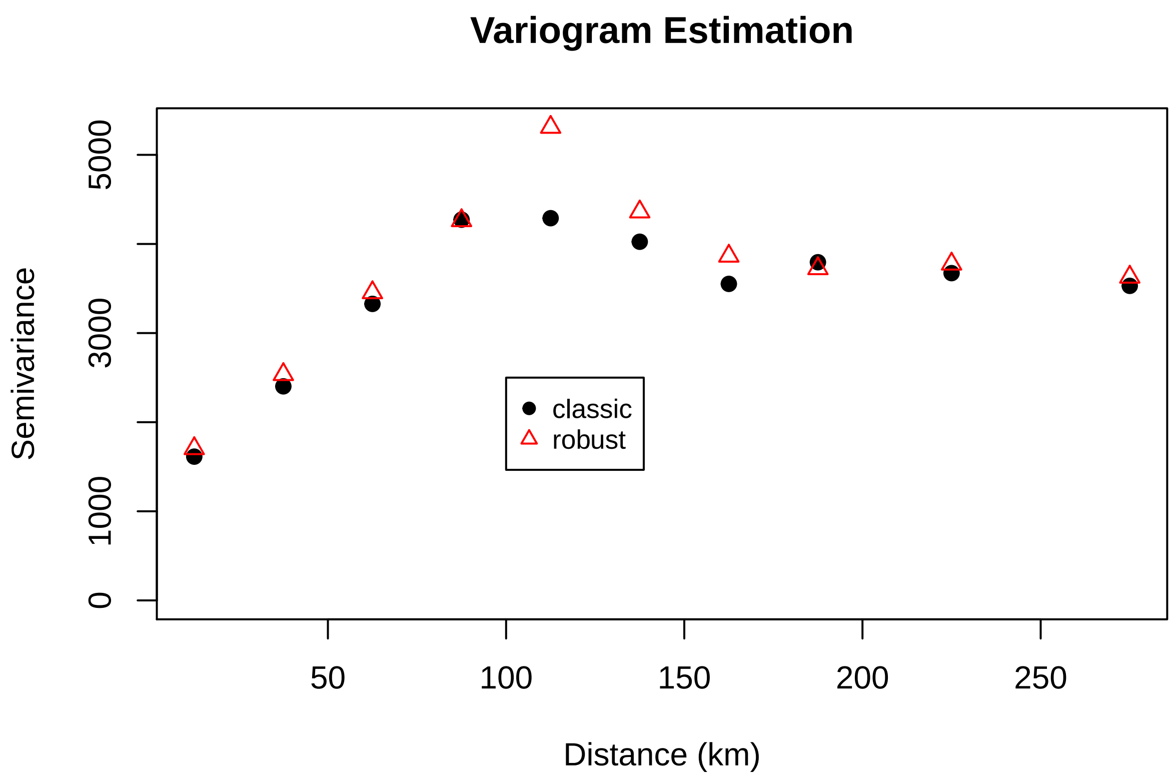
```r
gamma.bar = variog(wolfcamp, breaks = breaks, estimator.type = "modulus")
```

```
## variog: computing omnidirectional variogram
```

```r
gamma.hat$n
```

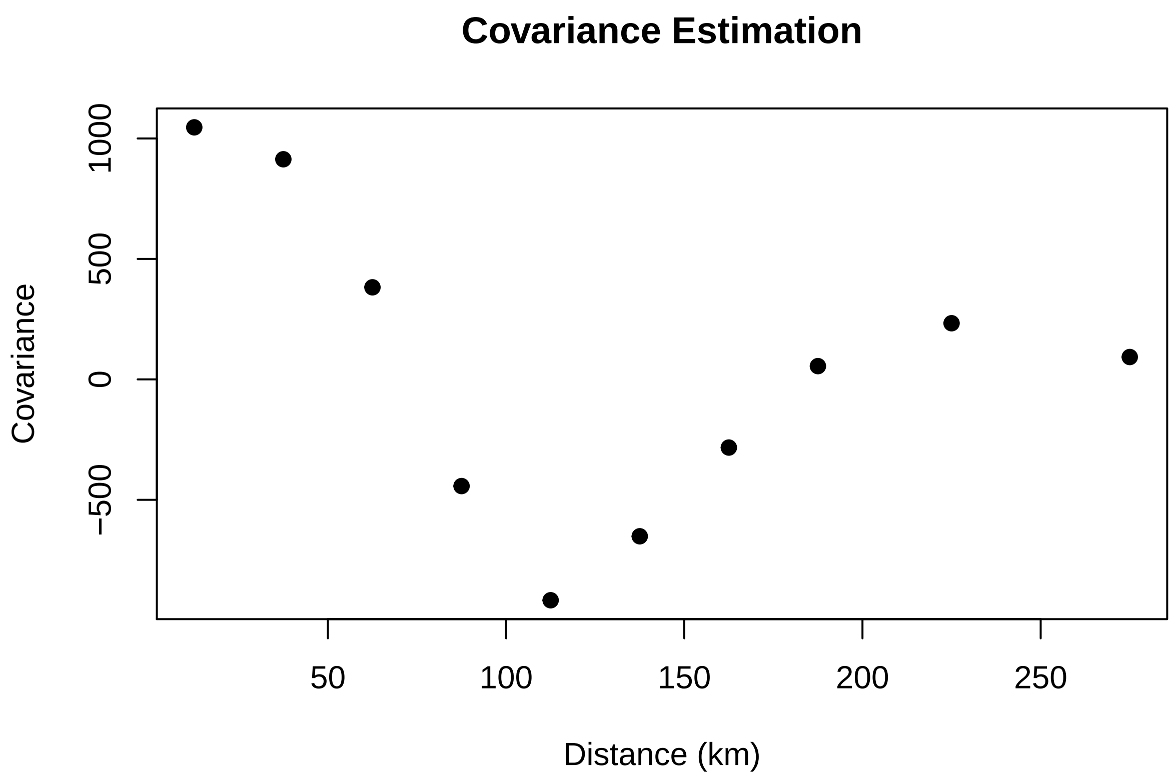
```
##  [1] 110 213 205 243 292 347 451 391 589 448
```

```r
plot(x = gamma.hat$u, y = gamma.hat$v, pch = 19, ylim = c(0, max(gamma.hat$v,
  gamma.bar$v)), main = "Variogram Estimation", xlab = "Distance (km)", ylab =
  "Semivariance")
points(x = gamma.bar$u, y = gamma.bar$v, pch = 2, col = "red")
legend(100, 2500, pch = c(19,2), c("classic", "robust"), cex = 0.8, col = c("black",
  "red"))
```

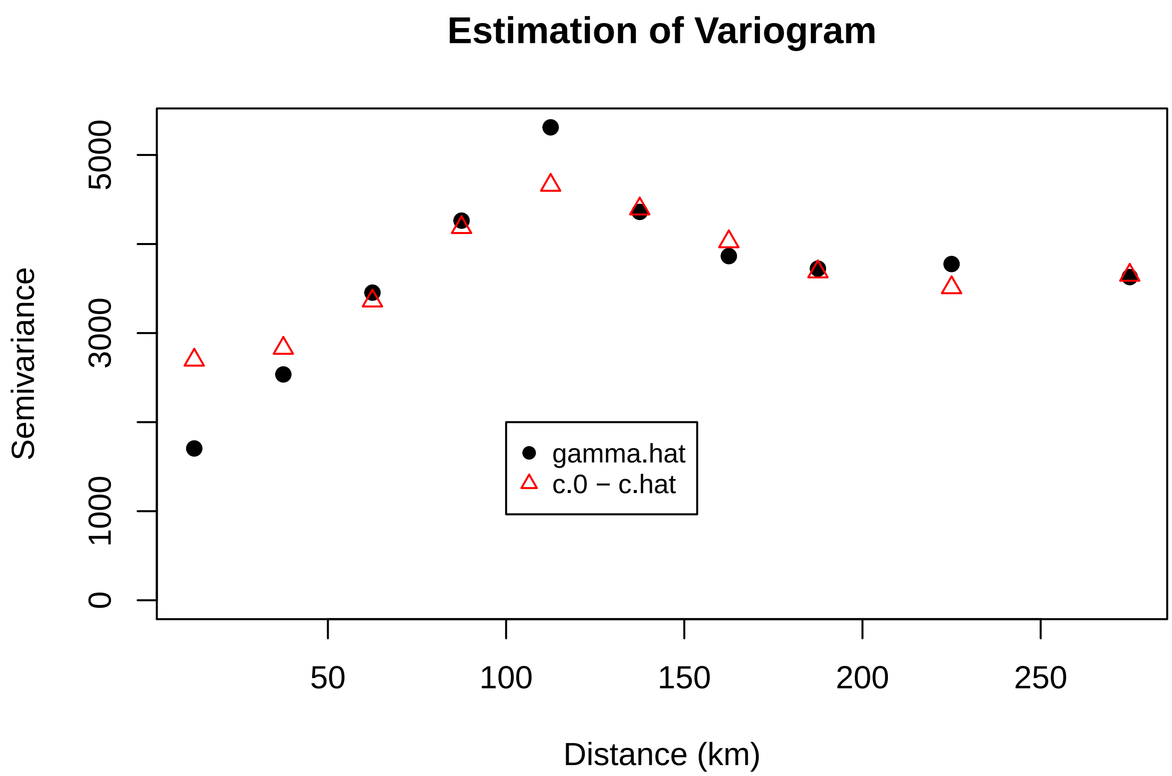



```r
  # Empirical covariance

get.points.indices = function(x, y, min.dist, max.dist)
{
  pts.1 = NULL
  pts.2 = NULL

  for (i in 1:length(x))
  {
    for (j in i:length(x))
```

```r
    {
      dist = sqrt((x[i] - x[j])**2 + (y[i] - y[j])**2)
      if (min.dist < dist & dist <= max.dist)
      {
        pts.1 = c(pts.1, i)
        pts.2 = c(pts.2, j)
      }
    }
  }

  return(cbind(pts.1, pts.2))
}

compute.c.hat = function(x, y, z, breaks)
{
  if (breaks[1] > 0)
    breaks = c(0, breaks)

  n = NULL
  c = NULL

  for (i in 1:(length(breaks)-1))
  {
    points = get.points.indices(x, y, breaks[i], breaks[i+1])
    n = c(n, length(points[,1]))
    c = c(c, sum(z[points[,1]] * z[points[,2]]) / n[i])
  }

  u = NULL
  for (i in 1:(length(breaks)-1))
    u = c(u, (breaks[i] + breaks[i+1]) / 2)

  ret = list(u = u, c = c, n = n)
  return(ret)
}

# Compute & plot covariance

c.hat = compute.c.hat(x, y, wolfcamp$data, breaks)
plot(x = c.hat$u, y = c.hat$c, pch = 19, ylim = c(min(c.hat$c), max(c.hat$c)), main =
  "Covariance Estimation", xlab = "Distance (km)", ylab = "Covariance")
```




```r
# Compute & plot methods of estimating variogram

c.0 = sum(wolfcamp$data**2) / length(wolfcamp$data)
plot(x = gamma.bar$u, y = gamma.bar$v, pch = 19, ylim = c(0, max(gamma.bar$v, c.0 -
  c.hat$c)), main = "Estimation of Variogram", xlab = "Distance (km)", ylab = "Semivariance")
points(x = c.hat$u, y = c.0 - c.hat$c, pch = 2, col = "red")
legend(100, 2000, pch = c(19,2), c("gamma.hat", "c.0 - c.hat"), cex = 0.8, col =
  c("black", "red"))
```




```
####################################
# Directional Variogram & Covariance
####################################

# Compute directional variogram

par(mfrow = c(1,1))
gamma.hat.a0 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = 0)
```

```
## variog: computing variogram for direction = 0 degrees (0 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
```

```
gamma.hat.a45 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = pi/4)
```

```
## variog: computing variogram for direction = 45 degrees (0.785 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
```

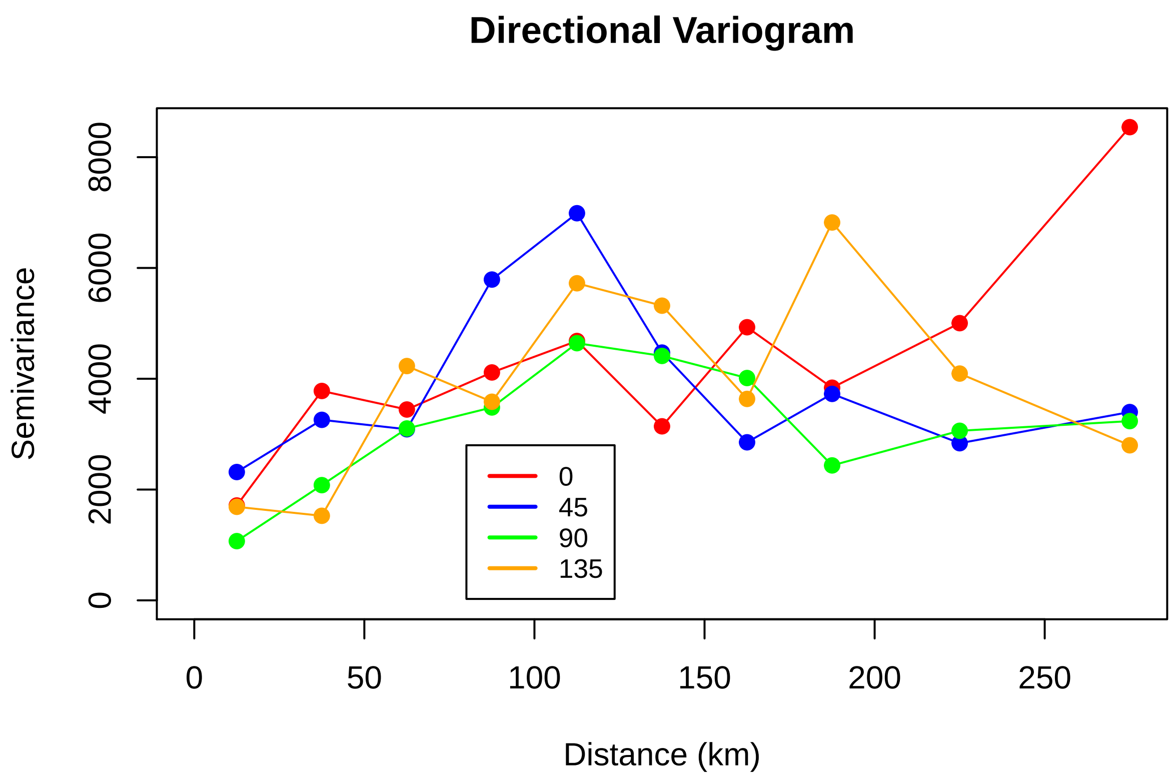
```
gamma.hat.a90 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = pi/2)
```

```
## variog: computing variogram for direction = 90 degrees (1.571 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
```

```
gamma.hat.a135 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = 3/4*pi)
```

```
## variog: computing variogram for direction = 135 degrees (2.356 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
```

```r
# Plot directional variogram

par(mfrow = c(1,1))
max.x = max(c(gamma.hat.a0$u, gamma.hat.a45$u, gamma.hat.a90$u, gamma.hat.a135$u))
min.y = min(c(gamma.hat.a0$v, gamma.hat.a45$v, gamma.hat.a90$v, gamma.hat.a135$v))
max.y = max(c(gamma.hat.a0$v, gamma.hat.a45$v, gamma.hat.a90$v, gamma.hat.a135$v))
plot(gamma.hat.a0, col = "red", pch = 19, type = "o", xlim = c(0, max.x), ylim =
  c(0, max.y), main = "Directional Variogram", xlab = "Distance (km)", ylab =
  "Semivariance")
lines(gamma.hat.a45, col = "blue", pch = 19, type = "o")
lines(gamma.hat.a90, col = "green", pch = 19, type = "o")
lines(gamma.hat.a135, col = "orange", pch = 19, type = "o")
legend(80, 2800, col = c("red", "blue", "green", "orange"), c(0, 45, 90, 135), lwd =
  c(2, 2, 2, 2), cex = 0.8)
```




```r
# Effect of decreasing tolerance angle

gamma.hat.a0 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = 0, tolerance = pi/16)
```

```
## variog: computing variogram for direction = 0 degrees (0 radians)
##         tolerance angle = 11.25 degrees (0.196 radians)
```

```r
gamma.hat.a45 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = pi/4, tolerance = pi/16)
```

```
## variog: computing variogram for direction = 45 degrees (0.785 radians)
##         tolerance angle = 11.25 degrees (0.196 radians)
```

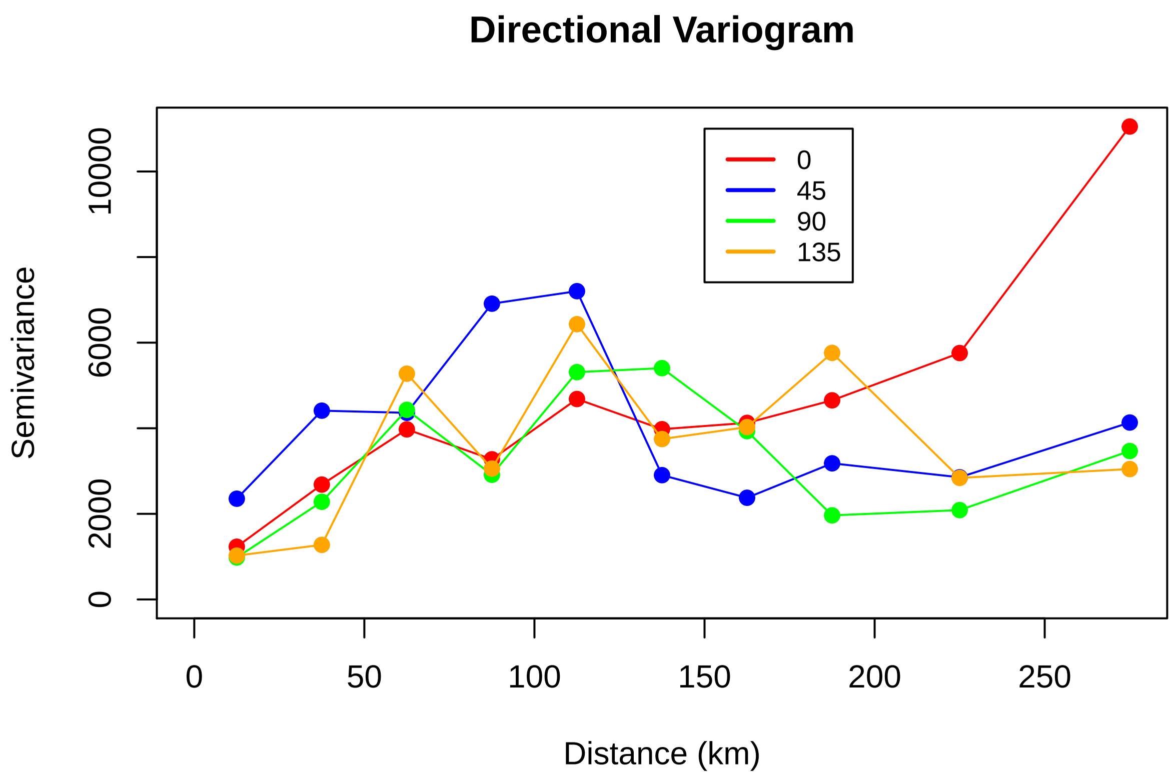
```r
gamma.hat.a90 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = pi/2, tolerance = pi/16)
```

```
## variog: computing variogram for direction = 90 degrees (1.571 radians)
##         tolerance angle = 11.25 degrees (0.196 radians)
```

```
gamma.hat.a135 = variog(wolfcamp, estimator.type = "modulus", breaks = breaks,
  direction = 3/4*pi, tolerance = pi/16)
```

```
## variog: computing variogram for direction = 135 degrees (2.356 radians)
##         tolerance angle = 11.25 degrees (0.196 radians)
```

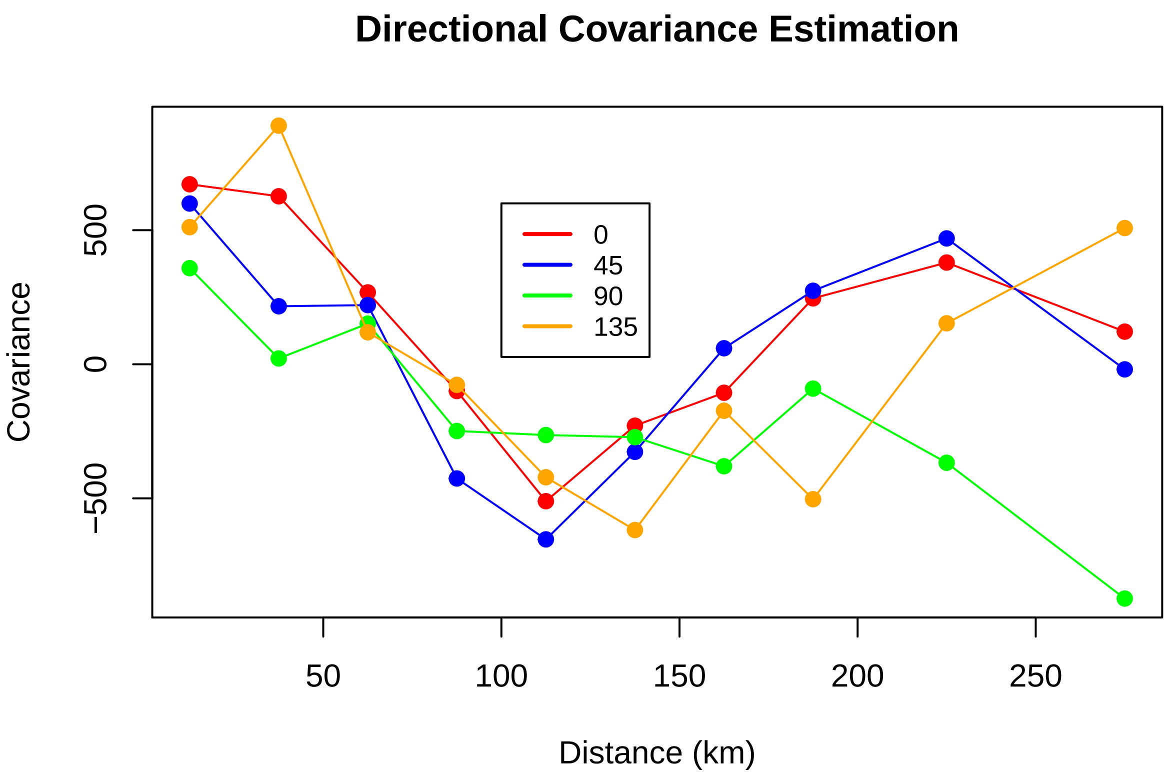
```
max.x = max(c(gamma.hat.a0$u, gamma.hat.a45$u, gamma.hat.a90$u, gamma.hat.a135$u))
min.y = min(c(gamma.hat.a0$v, gamma.hat.a45$v, gamma.hat.a90$v, gamma.hat.a135$v))
max.y = max(c(gamma.hat.a0$v, gamma.hat.a45$v, gamma.hat.a90$v, gamma.hat.a135$v))
plot(gamma.hat.a0, col = "red", pch = 19, type = "o", xlim = c(0, max.x), ylim =
  c(0, max.y), main = "Directional Variogram", xlab = "Distance (km)", ylab =
  "Semivariance")
lines(gamma.hat.a45, col = "blue", pch = 19, type = "o")
lines(gamma.hat.a90, col = "green", pch = 19, type = "o")
lines(gamma.hat.a135, col = "orange", pch = 19, type = "o")
legend(150, 11000, col = c("red", "blue", "green", "orange"), c(0, 45, 90, 135),
  lwd = c(2, 2, 2, 2), cex = 0.8)
```




```
# Manual function for directional covariance

get.points.indices.directional = function(x, y, min.dist, max.dist, angle,
  tolerance = pi/8)
{
  pts.1 = NULL
  pts.2 = NULL

  for (i in 1:(length(x)-1))
  {
```

```
    for (j in (i+1):length(x))
    {
      dist = sqrt((x[i] - x[j])**2 + (y[i] - y[j])**2)
      ang = atan((y[j] - y[i]) / (x[j] - x[i]))
      if (ang < 0)
        ang = ang + pi
      if (angle - tolerance < 0)
        angle.cond = (angle - tolerance <= ang & ang <= angle + tolerance) |
          (angle - tolerance + pi <= ang & ang <= angle + tolerance + pi)
      else if (angle + tolerance > pi)
        angle.cond = (angle - tolerance <= ang & ang <= angle + tolerance) |
          (angle - tolerance - pi <= ang & ang <= angle + tolerance - pi)
      else
        angle.cond = angle - tolerance <= ang & ang <= angle + tolerance
      if (angle.cond & min.dist < dist & dist <= max.dist)
      {
        pts.1 = c(pts.1, i)
        pts.2 = c(pts.2, j)
      }
    }
  }

  return(cbind(pts.1, pts.2))
}

compute.c.hat.directional = function(x, y, z, breaks, angle, tolerance = pi/8,
  angle.type = "radian")
{
  if (angle.type == "degree")
  {
    angle = angle * pi / 180
    tolerance = tolerance * pi / 180
  }

  if (breaks[1] > 0)
    breaks = c(0, breaks)

  n = NULL
  c = NULL

  for (i in 1:(length(breaks)-1))
  {
    points = get.points.indices.directional(x, y, breaks[i], breaks[i+1], angle,
      tolerance)
    n = c(n, length(points[,1]))
    c = c(c, sum(z[points[,1]] * z[points[,2]]) / n[i] / 2)
  }

  u = NULL
  for (i in 1:(length(breaks)-1))
    u = c(u, (breaks[i] + breaks[i+1]) / 2)

  ret = list(u = u, c = c, n = n)
```

```
  return(ret)
}

# Compute directional covariance

c.hat.a0 = compute.c.hat.directional(x, y, wolfcamp$data, breaks, 0)
c.hat.a45 = compute.c.hat.directional(x, y, wolfcamp$data, breaks, pi/4)
c.hat.a90 = compute.c.hat.directional(x, y, wolfcamp$data, breaks, pi/2)
c.hat.a135 = compute.c.hat.directional(x, y, wolfcamp$data, breaks, pi*3/4)

# Plot directional covariance

min.val = min(na.omit(c(c.hat.a0$c, c.hat.a45$c, c.hat.a90$c, c.hat.a135$c)))
max.val = max(na.omit(c(c.hat.a0$c, c.hat.a45$c, c.hat.a90$c, c.hat.a135$c)))
plot(x = c.hat.a0$u, y = c.hat.a0$c, pch = 19, type = "o", col = "red", ylim =
  c(min.val, max.val), main = "Directional Covariance Estimation", xlab = "Distance (km)",
  ylab = "Covariance")
lines(x = c.hat.a45$u, y = c.hat.a45$c, pch = 19, type = "o", col = "blue")
lines(x = c.hat.a90$u, y = c.hat.a90$c, pch = 19, type = "o", col = "green")
lines(x = c.hat.a135$u, y = c.hat.a135$c, pch = 19, type = "o", col = "orange")
legend(100, 600, col = c("red", "blue", "green", "orange"), c(0, 45, 90, 135), lwd =
  c(2, 2, 2, 2), cex = 0.8)
```




```
###################
# Variogram Fitting
###################

# Computes the sum of squared errors for the first three points for a given
# variogram
```

```r
sse3 = function(v, u, points)
{

  u = u[1:3]
  points = points[1:3]

  tausq = v$nugget
  sigmasq = v$cov.pars[1]
  phi = 1/v$cov.pars[2]

  if (v$cov.model == "linear")
    pred = tausq + sigmasq * u
  else
  {
    if (v$cov.model == "spherical")
      pred = tausq + sigmasq * (3/2*phi*u - 1/2*(phi*u)**3)
    else
    {
      if (v$cov.model == "matern")
      {
        nu = v$kappa
        pred = tausq + sigmasq * (1 - ((phi * u)**nu)/(2**(nu-1) * gamma(nu)) *
          besselK(phi * u, nu))
      }
      else
      {
        if (v$cov.model == "wave")
          pred = tausq + sigmasq * (1 - sin(phi * u) / (phi * u))
      }
    }
  }

  return(sum((points-pred)**2))

}

# Cressie-style weights

gamma.hat.lin = variofit(gamma.bar, cov.model = "linear", fix.nugget = FALSE, weights =
  "cressie")
```

```
## variofit: covariance model used is linear
## variofit: weights used: cressie
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq     kappa
## initial.value "3982.66" "0"   "2655.11" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 180.495668104337
```

```r
gamma.hat.sph = variofit(gamma.bar, cov.model = "spherical", fix.nugget = FALSE, weights =
  "cressie")
```

```
## variofit: covariance model used is spherical
## variofit: weights used: cressie
```

```
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq     kappa
## initial.value "2655.11" "88"  "1327.55" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 49.5154401042739
```

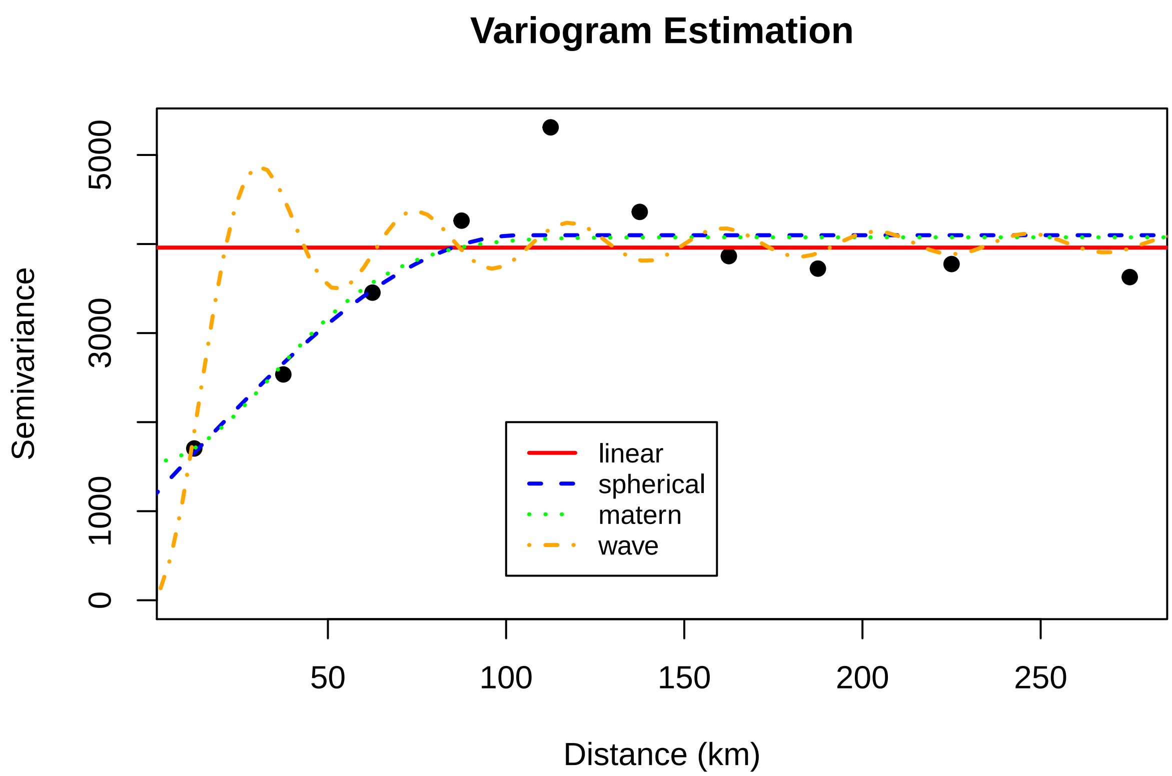
```r
gamma.hat.mat = variofit(gamma.bar, cov.model = "matern", fix.nugget = FALSE, kap = 1.5,
  fix.kappa = FALSE, weights = "cressie")
```

```
## variofit: covariance model used is matern
## variofit: weights used: cressie
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq kappa
## initial.value "3982.66" "44"  "0"   "0.25"
## status        "est"     "est" "est" "est"
## loss value: 66.1405408022394
```

```r
gamma.hat.wav = variofit(gamma.bar, cov.model = "wave", fix.nugget = FALSE, weights =
  "cressie")
```

```
## variofit: covariance model used is wave
## variofit: weights used: cressie
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq kappa
## initial.value "3982.66" "0"   "0"   "0.5"
## status        "est"     "est" "est" "fix"
## loss value: 111.687695012793
```

```r
plot(x = gamma.bar$u, y = gamma.bar$v, ylim = c(0, max(gamma.bar$v)), pch = 19, main =
  "Variogram Estimation", xlab = "Distance (km)", ylab = "Semivariance")
lines(gamma.hat.lin, col = "red", lwd = 2, lty = 1)
lines(gamma.hat.sph, col = "blue", lwd = 2, lty = 2)
lines(gamma.hat.mat, col = "green", lwd = 2, lty = 3)
lines(gamma.hat.wav, col = "orange", lwd = 2, lty = 4)
legend(100, 2000, col = c("red", "blue", "green", "orange"), c("linear", "spherical",
  "matern", "wave"), lty = c(1,2,3,4), cex = 0.8, lwd = 2)
```




```
sse3(gamma.hat.lin, gamma.bar$u, gamma.bar$v)
```

```
## [1] 7375242
```

```
sse3(gamma.hat.sph, gamma.bar$u, gamma.bar$v)
```

```
## [1] 19263.89
```

```
sse3(gamma.hat.mat, gamma.bar$u, gamma.bar$v)
```

```
## [1] 27655.26
```

```
sse3(gamma.hat.wav, gamma.bar$u, gamma.bar$v)
```

```
## [1] 4193901
```

```
# npairs weights

gamma.hat.lin = variofit(gamma.bar, cov.model = "linear", fix.nugget = FALSE, weights =
  "npairs")
```

```
## variofit: covariance model used is linear
## variofit: weights used: npairs
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##                 sigmasq   phi    tausq     kappa
## initial.value "2655.11" "0"    "2655.11" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 2223265315.2117
```

```
gamma.hat.sph = variofit(gamma.bar, cov.model = "spherical", fix.nugget = FALSE, weights =
  "npairs")
```

```
## variofit: covariance model used is spherical
## variofit: weights used: npairs
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq     kappa
## initial.value "2655.11" "88"  "1327.55" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 743331518.638569
```

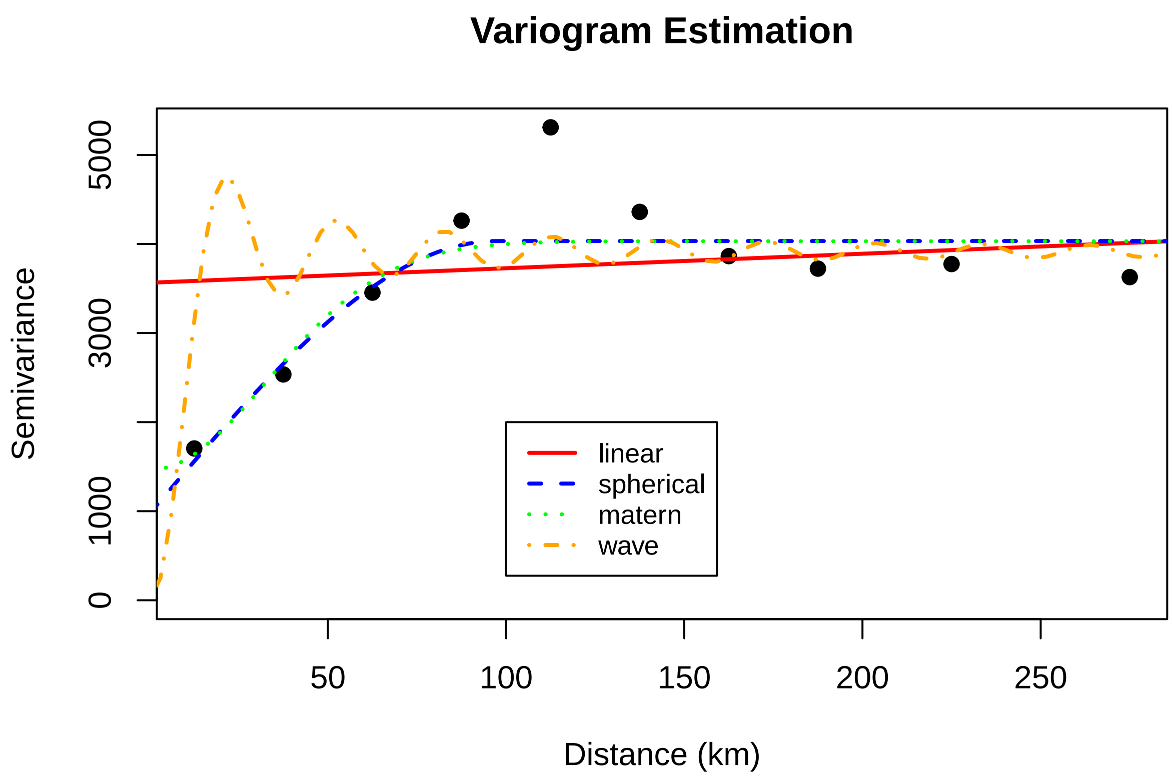
```
gamma.hat.mat = variofit(gamma.bar, cov.model = "matern", fix.nugget = FALSE, kap = 1.5,
  fix.kappa = FALSE, weights = "npairs")
```

```
## variofit: covariance model used is matern
## variofit: weights used: npairs
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq kappa
## initial.value "3982.66" "44"  "0"   "0.25"
## status        "est"     "est" "est" "est"
## loss value: 930429289.904966
```

```
gamma.hat.wav = variofit(gamma.bar, cov.model = "wave", fix.nugget = FALSE, weights =
  "npairs")
```

```
## variofit: covariance model used is wave
## variofit: weights used: npairs
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq kappa
## initial.value "3982.66" "0"   "0"   "0.5"
## status        "est"     "est" "est" "fix"
## loss value: 1771543383.49173
```

```
plot(x = gamma.bar$u, y = gamma.bar$v, ylim = c(0, max(gamma.bar$v)), pch = 19, main =
  "Variogram Estimation", xlab = "Distance (km)", ylab = "Semivariance")
lines(gamma.hat.lin, col = "red", lwd = 2, lty = 1)
lines(gamma.hat.sph, col = "blue", lwd = 2, lty = 2)
lines(gamma.hat.mat, col = "green", lwd = 2, lty = 3)
lines(gamma.hat.wav, col = "orange", lwd = 2, lty = 4)
legend(100, 2000, col = c("red", "blue", "green", "orange"), c("linear", "spherical",
  "matern", "wave"), lty = c(1,2,3,4), cex = 0.8, lwd = 2)
```




```r
sse3(gamma.hat.lin, gamma.bar$u, gamma.bar$v)
```

```
## [1] 4774109
```

```r
sse3(gamma.hat.sph, gamma.bar$u, gamma.bar$v)
```

```
## [1] 39309.48
```

```r
sse3(gamma.hat.mat, gamma.bar$u, gamma.bar$v)
```

```
## [1] 40557.29
```

```r
sse3(gamma.hat.wav, gamma.bar$u, gamma.bar$v)
```

```
## [1] 2976865
```

```r
# equal weights

gamma.hat.lin = variofit(gamma.bar, cov.model = "linear", fix.nugget = FALSE, weights =
  "equal")
```

```
## variofit: covariance model used is linear
## variofit: weights used: equal
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##                 sigmasq   phi   tausq     kappa
## initial.value "2655.11" "0"    "2655.11" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 7897271.1287311
```

```r
gamma.hat.sph = variofit(gamma.bar, cov.model = "spherical", fix.nugget = FALSE, weights =
  "equal")
```

```
## variofit: covariance model used is spherical
## variofit: weights used: equal
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq     kappa
## initial.value "2655.11" "88"  "1327.55" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 2467946.84838859
```

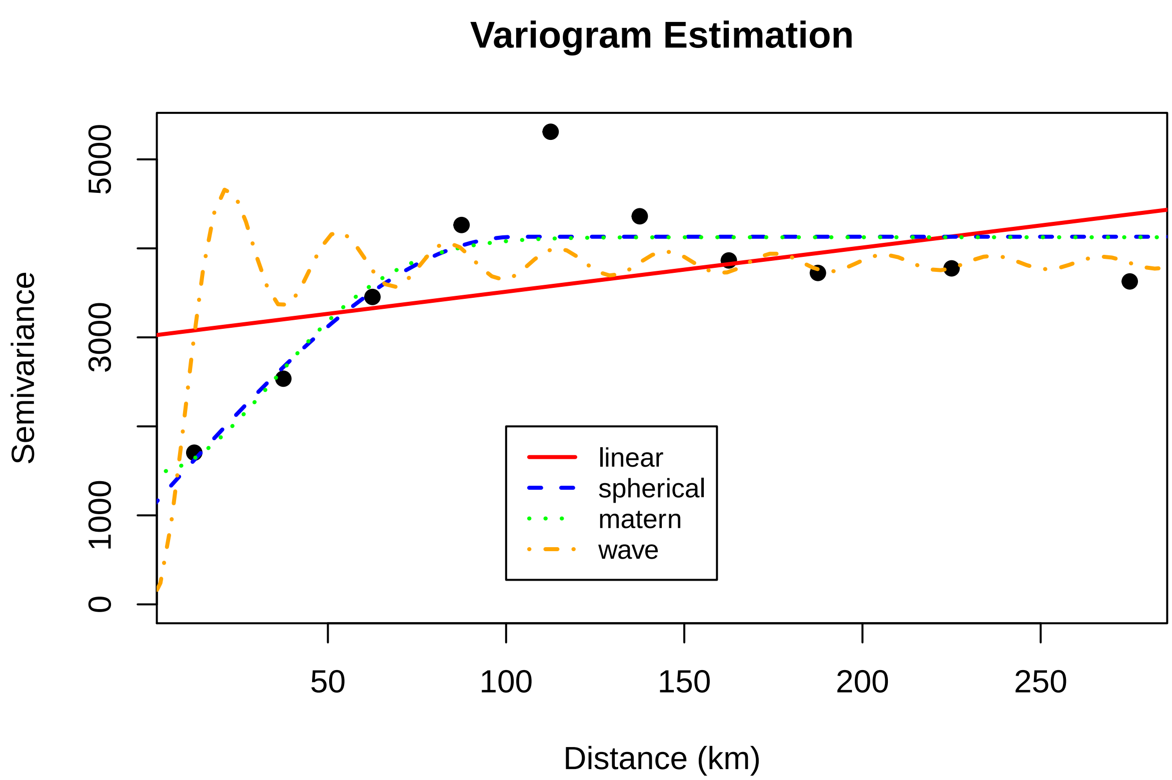
```
gamma.hat.mat = variofit(gamma.bar, cov.model = "matern", fix.nugget = FALSE, kap = 1.5,
  fix.kappa = FALSE, weights = "equal")
```

```
## variofit: covariance model used is matern
## variofit: weights used: equal
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq   phi   tausq kappa
## initial.value "3982.66" "44"  "0"   "0.25"
## status        "est"     "est" "est" "est"
## loss value: 3173972.67459355
```

```
gamma.hat.wav = variofit(gamma.bar, cov.model = "wave", fix.nugget = FALSE,
  ini.cov.pars = c(3919, 4.8), weights = "equal")
```

```
## variofit: covariance model used is wave
## variofit: weights used: equal
## variofit: minimisation function used: optim
```

```
plot(x = gamma.bar$u, y = gamma.bar$v, ylim = c(0, max(gamma.bar$v)), pch = 19, main =
  "Variogram Estimation", xlab = "Distance (km)", ylab = "Semivariance")
lines(gamma.hat.lin, col = "red", lwd = 2, lty = 1)
lines(gamma.hat.sph, col = "blue", lwd = 2, lty = 2)
lines(gamma.hat.mat, col = "green", lwd = 2, lty = 3)
lines(gamma.hat.wav, col = "orange", lwd = 2, lty = 4)
legend(100, 2000, col = c("red", "blue", "green", "orange"), c("linear", "spherical",
  "matern", "wave"), lty = c(1,2,3,4), cex = 0.8, lwd = 2)
```




```
sse3(gamma.hat.lin, gamma.bar$u, gamma.bar$v)
```

```
## [1] 2350404
```

```
sse3(gamma.hat.sph, gamma.bar$u, gamma.bar$v)
```

```
## [1] 27458.41
```

```
sse3(gamma.hat.mat, gamma.bar$u, gamma.bar$v)
```

```
## [1] 35531.02
```

```
sse3(gamma.hat.wav, gamma.bar$u, gamma.bar$v)
```

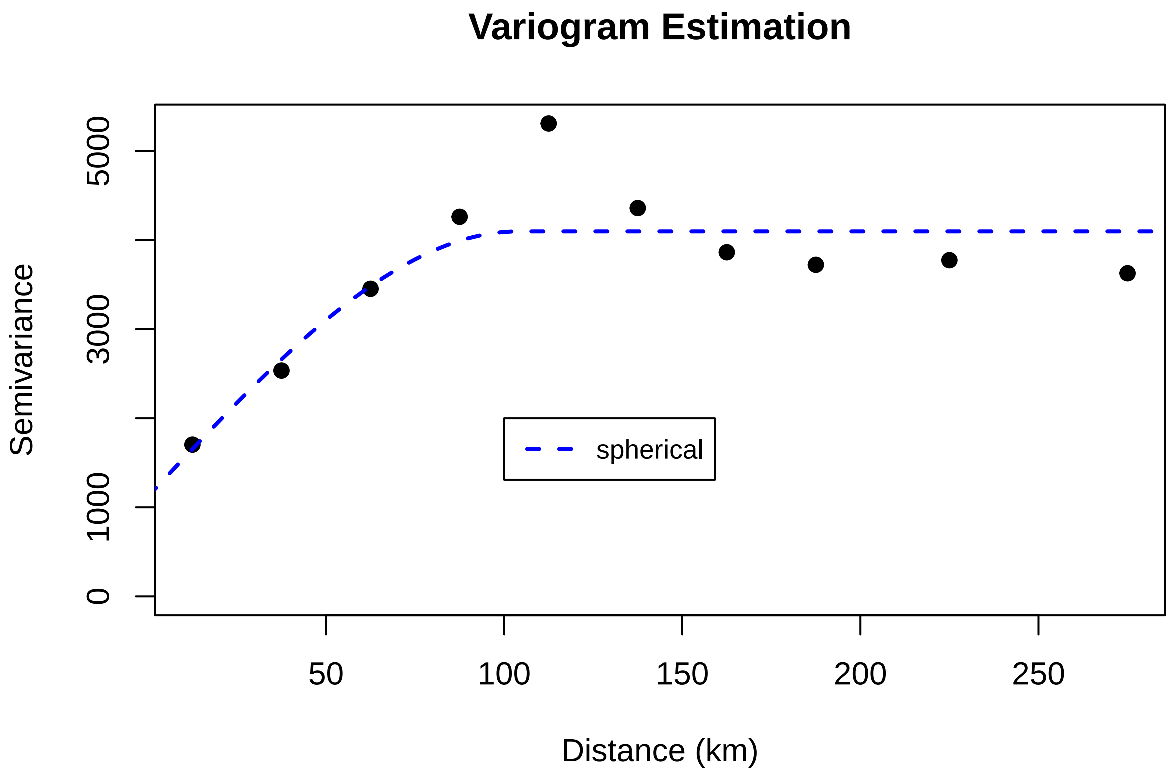
```
## [1] 2510289
```

```
  # Lowest sse3 - spherical variogram, Cressie-style weights, optim minimization
  # Try with nlm minimization

gamma.hat.sph = variofit(gamma.bar, cov.model = "spherical", fix.nugget = FALSE,
  minimisation.function = "nlm", weights = "cressie")
```

```
## variofit: covariance model used is spherical
## variofit: weights used: cressie
## variofit: minimisation function used: nlm
## variofit: searching for best initial value ... selected values:
##                 sigmasq   phi   tausq     kappa
## initial.value "2655.11" "88"  "1327.55" "0.5"
## status        "est"     "est" "est"     "fix"
## loss value: 49.5154401042739
```

```
plot(x = gamma.bar$u, y = gamma.bar$v, ylim = c(0, max(gamma.bar$v)), pch = 19, main =
  "Variogram Estimation", xlab = "Distance (km)", ylab = "Semivariance")
```

```r
lines(gamma.hat.sph, col = "blue", lwd = 2, lty = 2)
legend(100, 2000, col = "blue", "spherical", lty = 2, cex = 0.8, lwd = 2)
```




```r
sse3(gamma.hat.sph, gamma.bar$u, gamma.bar$v)
```

```
## [1] 19263.85
```

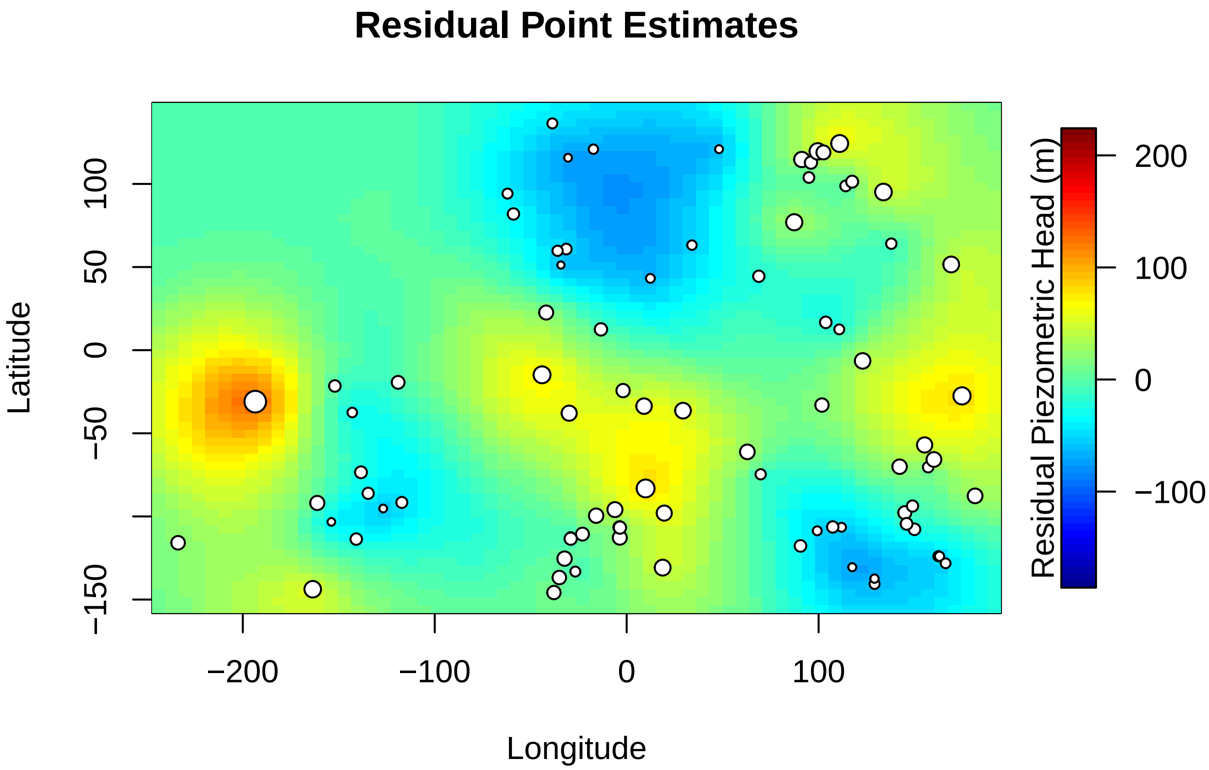
```r
  # This sse3 is slightly lower, so this is the final variogram

# Checking parameter estimates
gamma.hat.sph
```

```
## variofit: model parameters estimated by WLS (weighted least squares):
## covariance model is: spherical
## parameter estimates:
##     tausq   sigmasq       phi
## 1119.7552 2978.8657  103.9361
## Practical Range with cor=0.05 for asymptotic range: 103.9361
##
## variofit: minimised weighted sum of squares = 42.9362
```

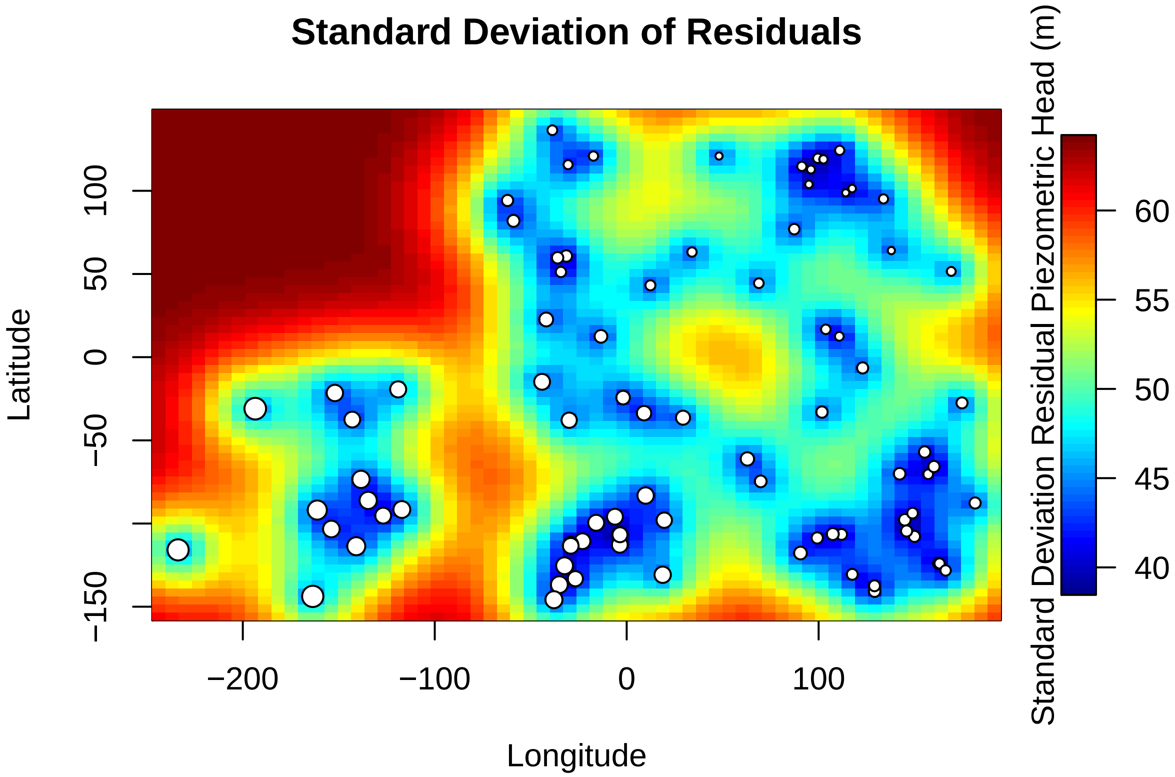
```r
#########
# Kriging
#########

kc = krige.control(type = "sk", obj.model = gamma.hat.sph)
sk = krige.conv(wolfcamp, locations = grid, krige = kc)
```

```
## krige.conv: model with constant mean
## krige.conv: Kriging performed using global neighbourhood
```

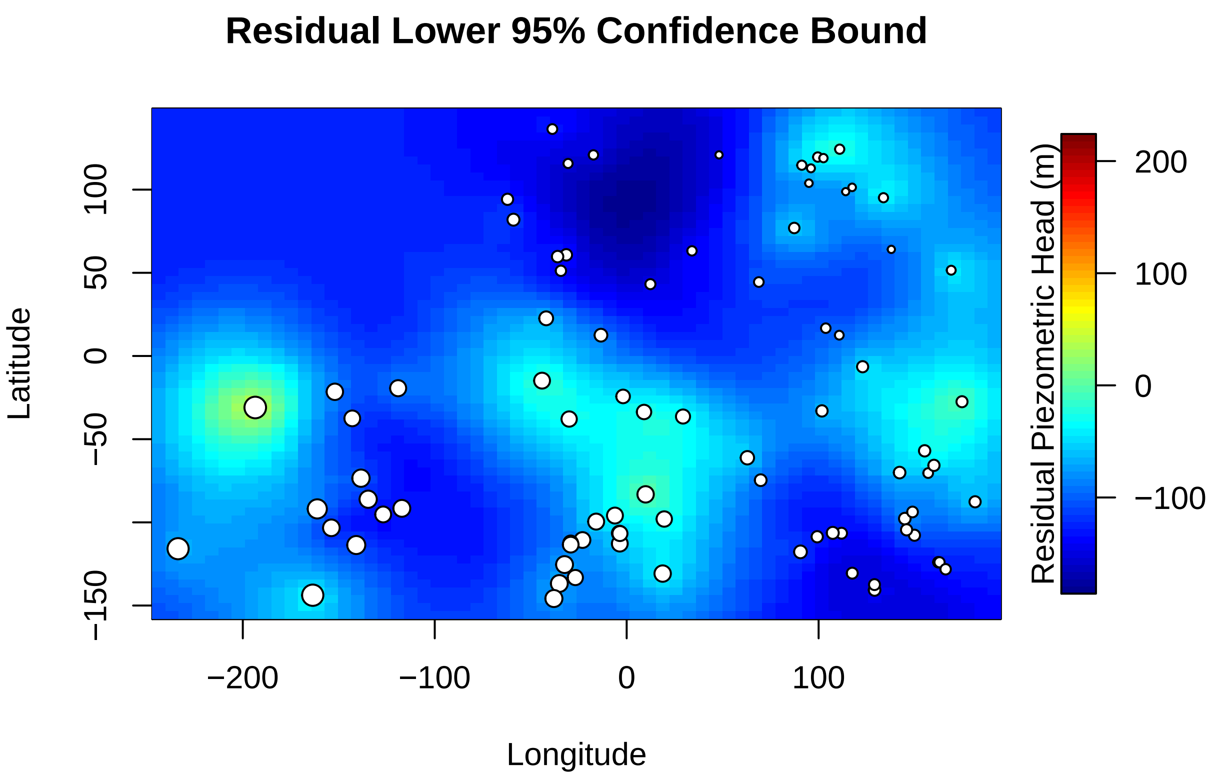
```r
  # Fixing scale for plots
min.val = min(sk$predict - qnorm(0.975) * sqrt(sk$krige.var))
max.val = max(sk$predict + qnorm(0.975) * sqrt(sk$krige.var))

# Plotting point estimates of residuals

quilt.plot(grid, sk$predict, zlim = c(min.val, max.val), main =
  "Residual Point Estimates", xlab = "Longitude", ylab = "Latitude", legend.args =
  list(text = "Residual Piezometric Head (m)", side = 2))
points(wolfcamp, pch = 21, col = "white", add = TRUE)
```



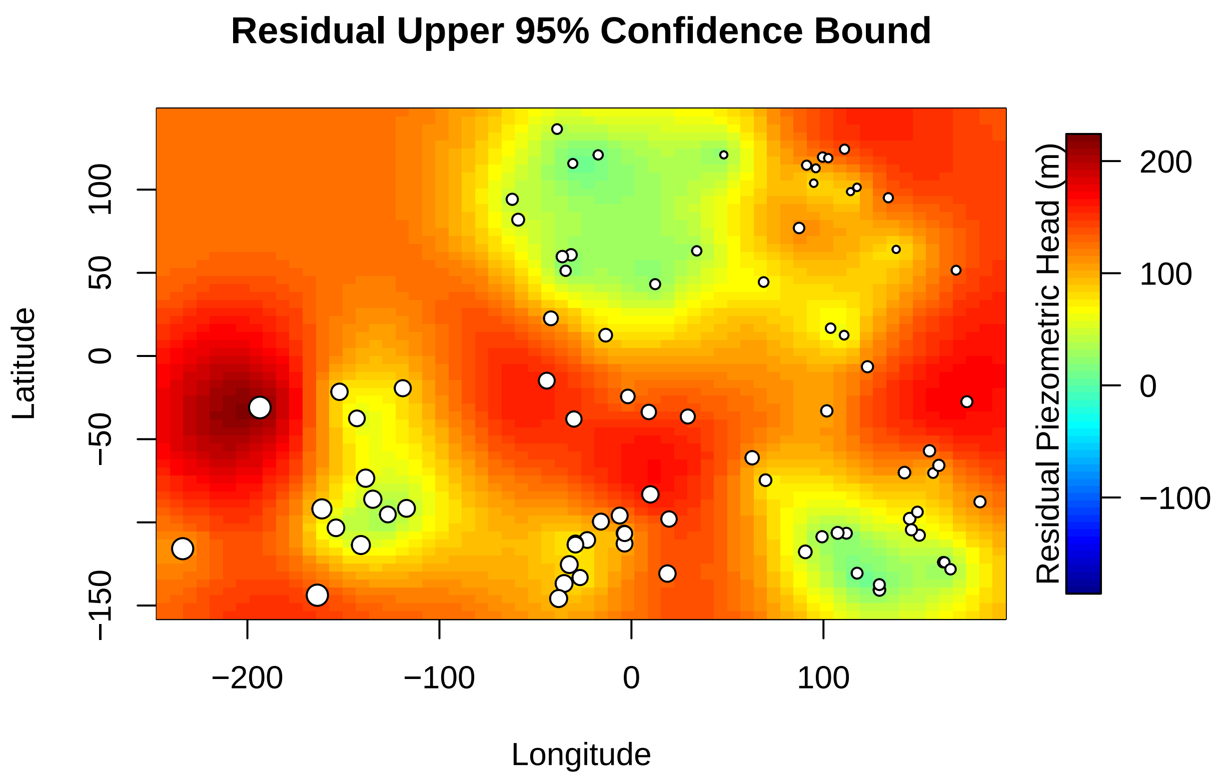

```r
# Plotting the kriging standard deviation of residuals

quilt.plot(grid, sqrt(sk$krige.var), main = "Standard Deviation of Residuals", xlab =
  "Longitude", ylab = "Latitude", legend.args = list(text =
  "Standard Deviation Residual Piezometric Head (m)", side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```



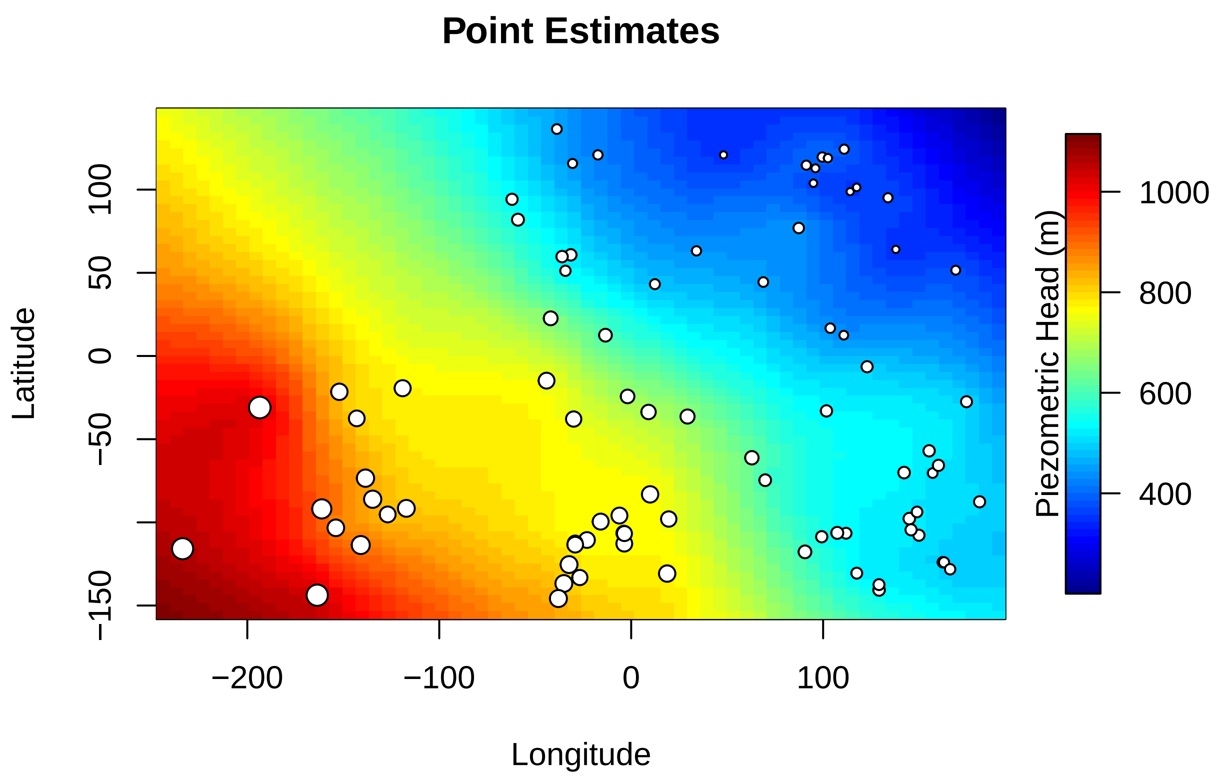

```
# Plotting lower 95% confidence estimates of residuals

quilt.plot(grid, sk$predict - qnorm(0.975) * sqrt(sk$krige.var), zlim =
  c(min.val, max.val), main = "Residual Lower 95% Confidence Bound", xlab = "Longitude",
  ylab = "Latitude", legend.args = list(text = "Residual Piezometric Head (m)", side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```



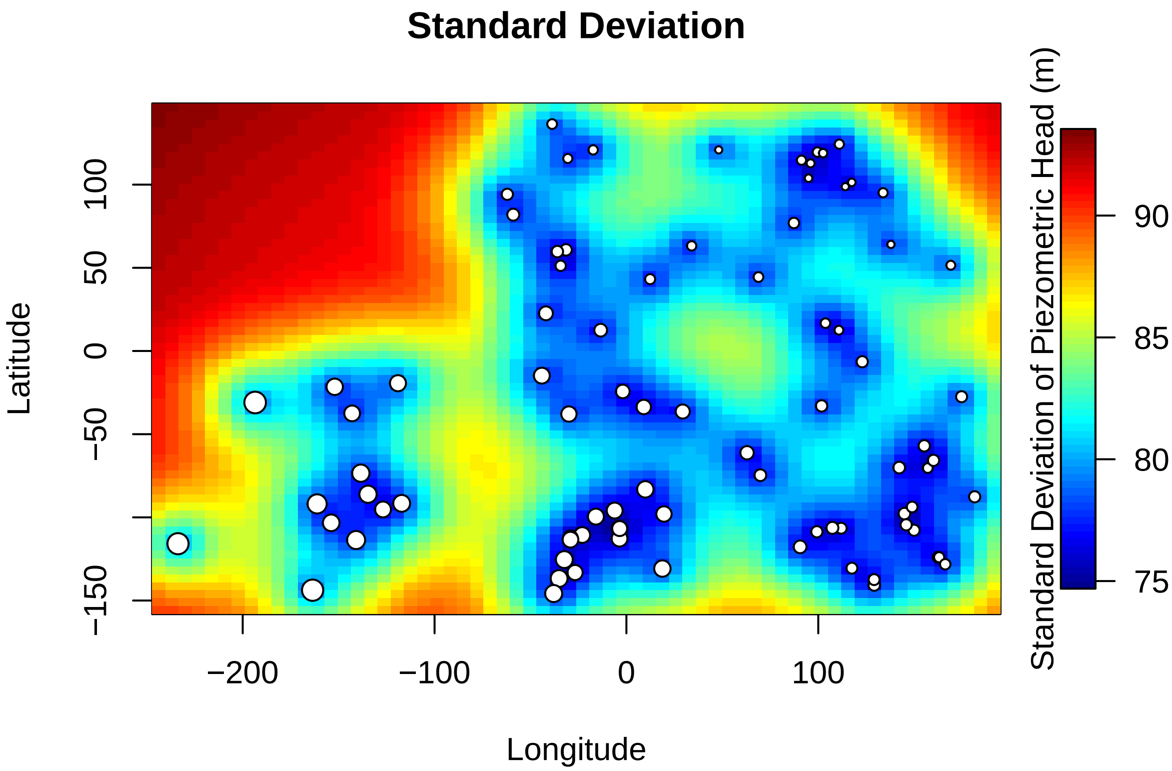

```r
# Plotting upper 95% confidence estimates of residuals

quilt.plot(grid, sk$predict + qnorm(0.975) * sqrt(sk$krige.var), zlim =
  c(min.val, max.val), main = "Residual Upper 95% Confidence Bound", xlab = "Longitude",
  ylab = "Latitude", legend.args = list(text = "Residual Piezometric Head (m)", side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```



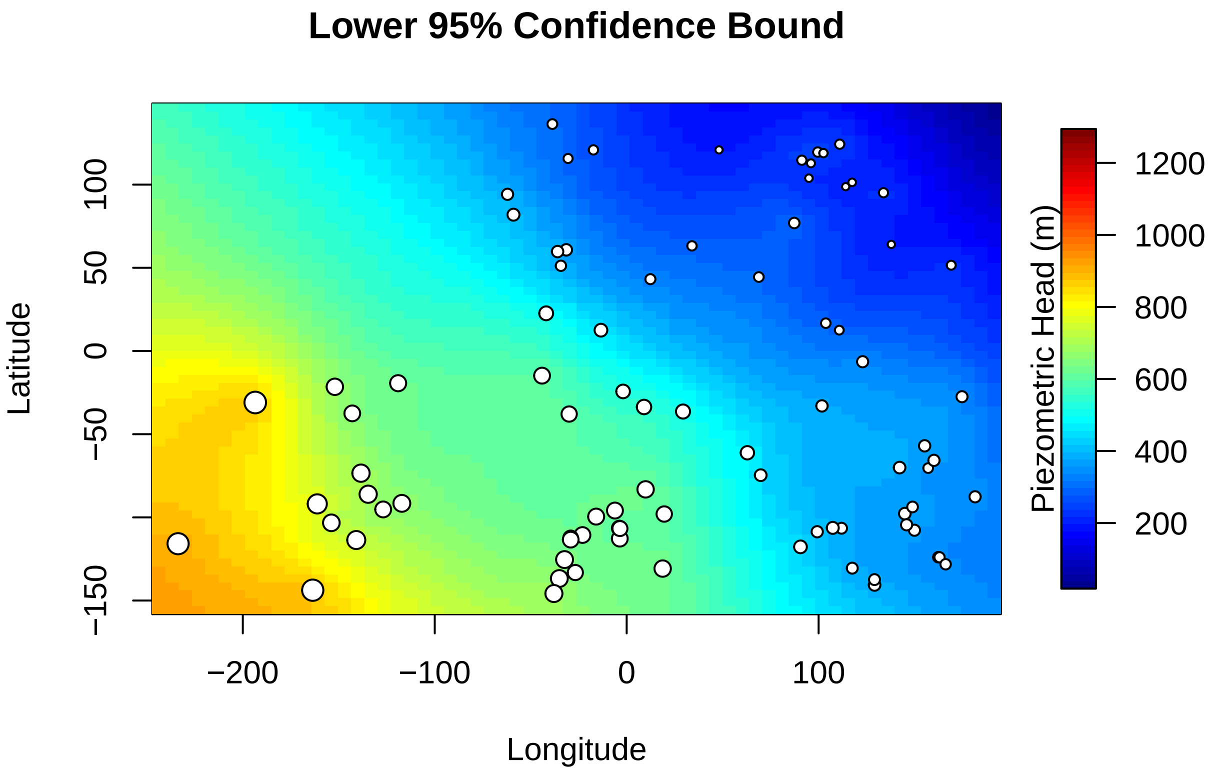

```r
  # Fixing scale for plots
min.val = min((pred.point + sk$predict) - (qnorm(0.975) * sqrt(sk$krige.var +
  pred.sd**2)))
max.val = max((pred.point + sk$predict) + (qnorm(0.975) * sqrt(sk$krige.var +
  pred.sd**2)))

# Plotting point estimates on the original scale

quilt.plot(grid, pred.point + sk$predict, main = "Point Estimates", xlab = "Longitude",
  ylab = "Latitude", legend.args = list(text = "Piezometric Head (m)", side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```



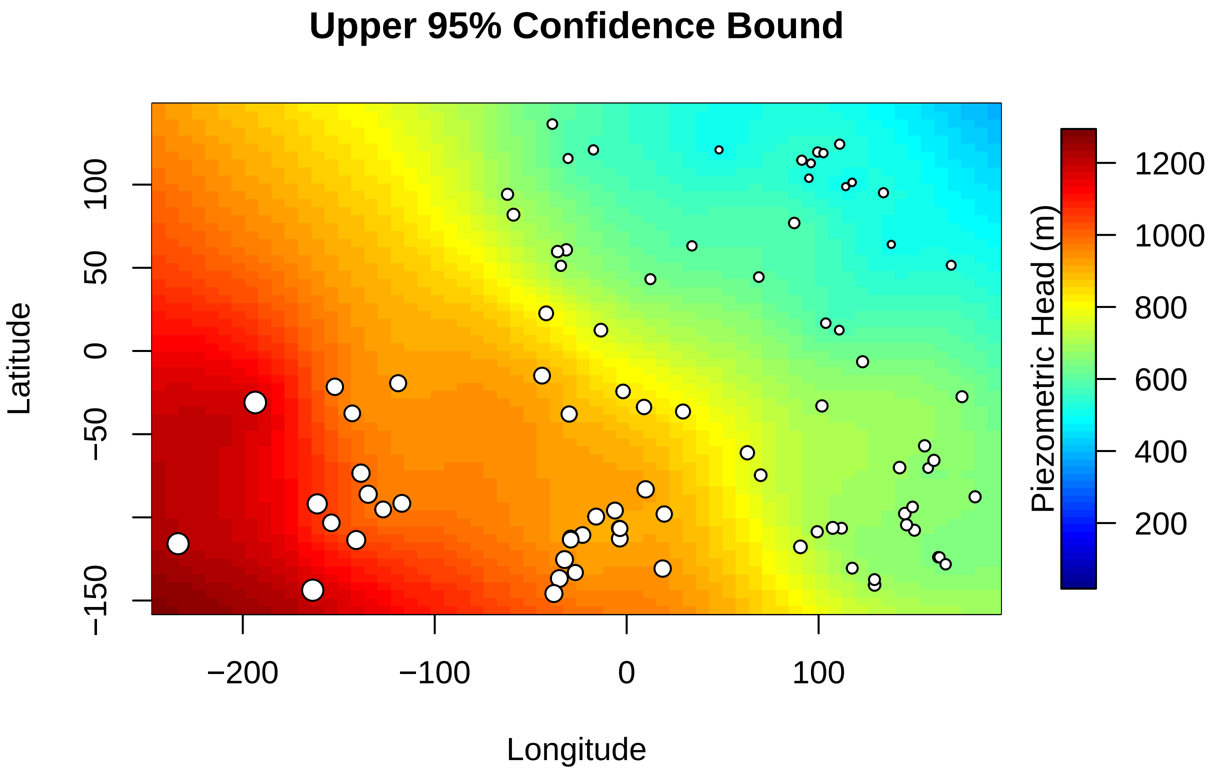

```
# Plotting the kriging standard deviation on the original scale

quilt.plot(grid, sqrt(pred.sd**2 + sk$krige.var), main = "Standard Deviation",
  xlab = "Longitude", ylab = "Latitude", legend.args = list(text =
  "Standard Deviation of Piezometric Head (m)", side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```



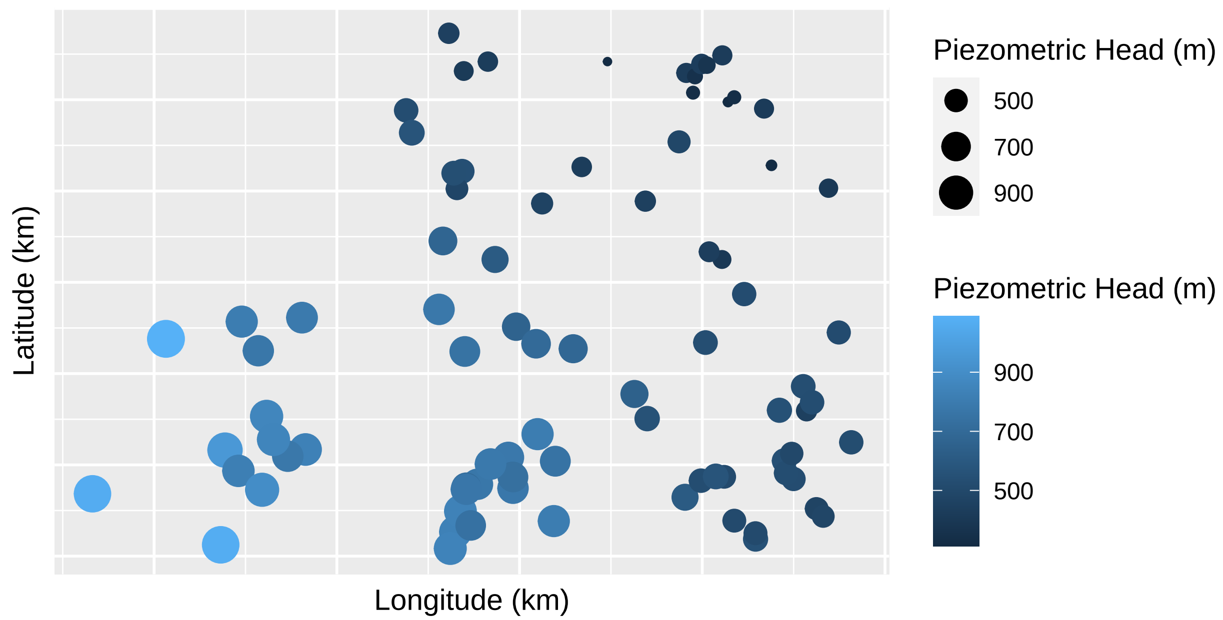

```r
# Plotting lower 95% confidence estimates on original scale

quilt.plot(grid, (pred.point + sk$predict) - (qnorm(0.975) * sqrt(sk$krige.var +
  pred.sd**2)), zlim = c(min.val, max.val), main = "Lower 95% Confidence Bound", xlab =
  "Longitude", ylab = "Latitude", legend.args = list(text = "Piezometric Head (m)",
  side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```



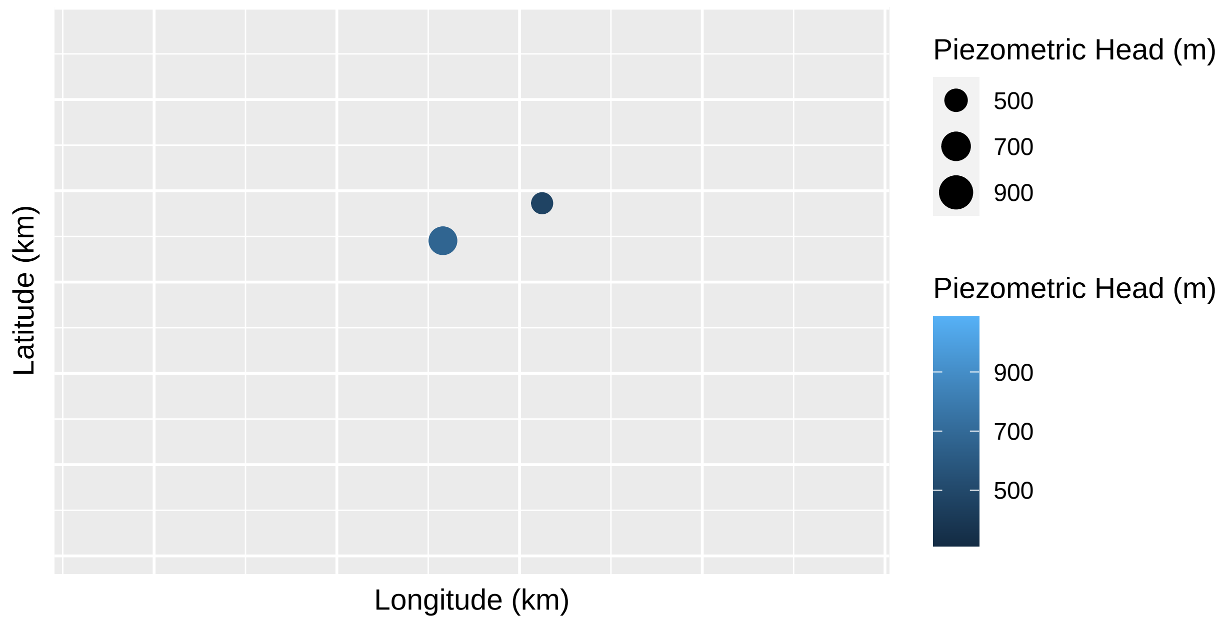

```r
# Plotting upper 95% confidence estimates on original scale

quilt.plot(grid, (pred.point + sk$predict) + (qnorm(0.975) * sqrt(sk$krige.var +
  pred.sd**2)), zlim = c(min.val, max.val), main = "Upper 95% Confidence Bound", xlab =
  "Longitude", ylab = "Latitude", legend.args = list(text = "Piezometric Head (m)",
  side = 2))
points(wolfcamp.orig, pch = 21, col = "white", add = TRUE)
```




```r
####################
# Pressure Difference
####################

# Plot of two points
df = data.frame(lat = y, long = x, Z = z)
p = ggplot() + geom_point(data = df, aes(x = long, y = lat, size = Z, color = Z))
p = p + labs(x = "Longitude (km)", y = "Latitude (km)", size = "Piezometric Head (m)",
  colour = "Piezometric Head (m)") + coord_fixed()
p = p + theme(axis.text.x = element_blank(), axis.text.y = element_blank(), axis.ticks =
  element_blank())
p
```




```
df = df[c(15,59),]
p = ggplot() + geom_point(data = df, aes(x = long, y = lat, size = Z, color = Z))
p = p + labs(x = "Longitude (km)", y = "Latitude (km)", size = "Piezometric Head (m)",
  colour = "Piezometric Head (m)") + coord_fixed()
p = p + theme(axis.text.x = element_blank(), axis.text.y = element_blank(), axis.ticks =
  element_blank())
p = p + xlim(min(x), max(x)) + ylim(min(y), max(y))
p = p + expand_limits(z = min(z))
p = p + lims(size = c(min(z), max(z)), color = c(min(z), max(z)))
p
```




```
true.diff = wolfcamp.orig$dat[15] - wolfcamp.orig$dat[59]
true.diff
```

```
## [1] 205.1266
```

```r
loc1 = wolfcamp.orig$coords[15,]
loc2 = wolfcamp.orig$coords[59,]
locs = cbind(loc1, loc2)
sk = krige.conv(wolfcamp, locations = cbind(loc1, loc2), krige = kc)
```

```
## krige.conv: model with constant mean
## krige.conv: Kriging performed using global neighbourhood
```

```r
pred = predict(mod.z, newdata = data.frame(x = locs[,1], y = locs[,2]), interval =
  "prediction", level = 0.95)
pred.point = pred[,1]
pred.sd = (pred[,3] - pred[,1]) / qnorm(0.975)

pred.diff = (pred.point[1] + sk$predict[1]) - (pred.point[2] + sk$predict[2])
pred.sd = sqrt(pred.sd[1]**2 + sk$krige.var[1] + pred.sd[2]**2 + sk$krige.var[2])
pred.diff
```

```
##        1
## 202.4221
```

```r
pred.sd
```

```
##        1
## 111.9034
```

```r
p = 1 - pnorm(pred.diff / pred.sd)
p
```

```
##          1
## 0.03523322
```