Analyse von Geodaten

Dirk Seidensticker

7. Juli 2017

Grundlagen

Wichtige Pakete für die Arbeit mit Geodaten in R:

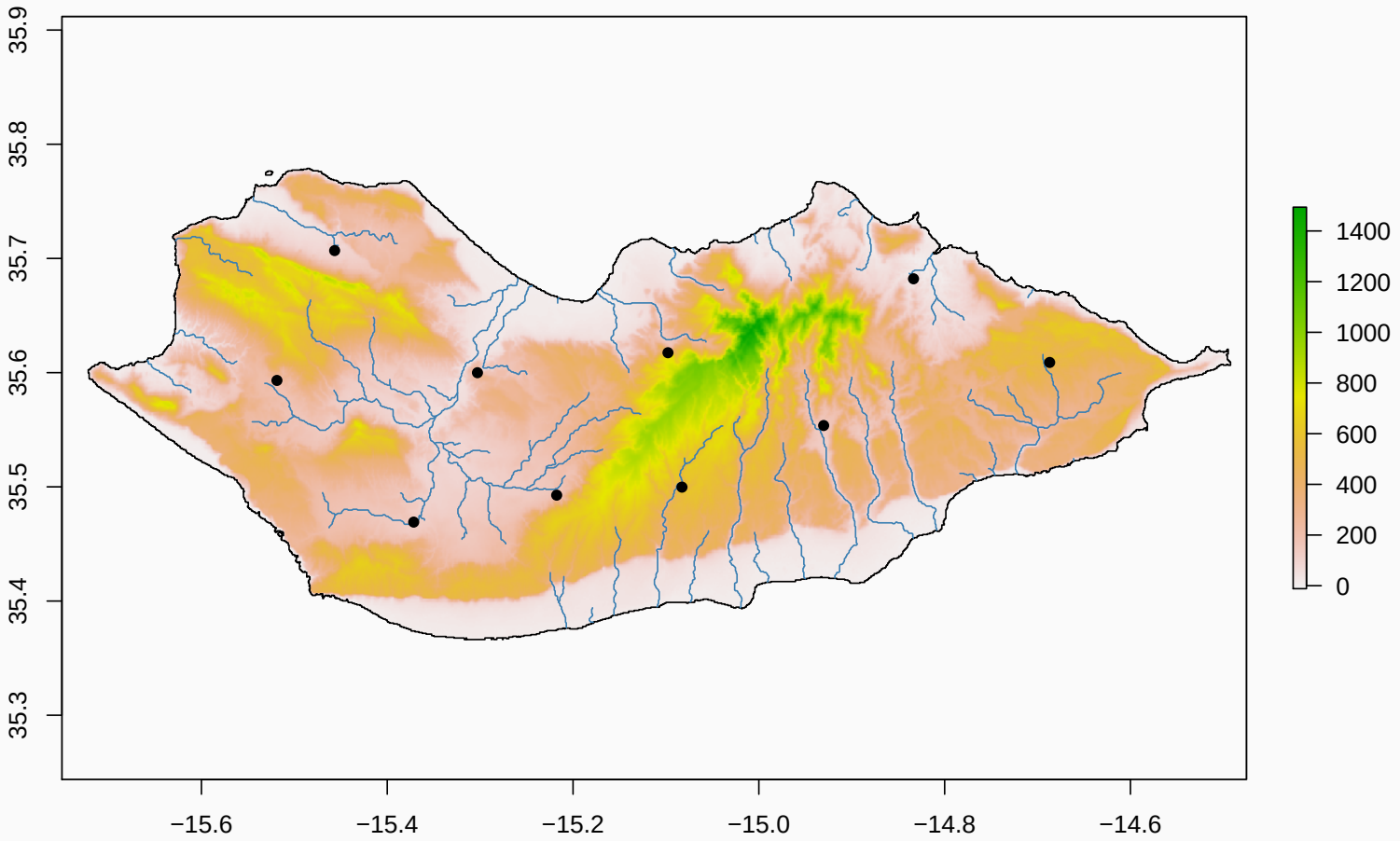
- `sp` - Hauptklassen für die Arbeit mit Geodaten
  - `spTransform()` - Änderung Projektion
- `rgdal` - R-Schnittstelle zu `gdal` (*Geospatial Data Abstraction Library*)
  - `readOGR()` - Einlesen von bspw. GeoJSON
- `rgeos` - R-Schnitstelle zu `geos` (*Geometry Engine Open Source*)
  - `gBuffer()` - zur Berechung von Buffern um Punkte
  - `gUnion()` - Führt Polygone zusammen
- `raster` - Klassen für die Arbeit mit Rasterdaten
  - `raster()` - Einlesen von Rasterdaten
  - `projectRaster()` - Änderung Projektion
  - `extract()` - Extrahiert Werte von Rasterobjekten
  - `disaggregate()` - Erstellung höher aufgelöster Raster-Daten
- `maptools` - Weitere Klassen für die Arbeit mit Geodaten

Siehe auch: *CRAN Task View: Analysis of Spatial Data*

Daten

- `raster/dem/atlantgis_dgm.tif` - Digitales Geländemodell
- `vector/geojson/sites.geojson` - Fundstellen
- `vector/geojson/coastline.geojson` - Küstenlinie
- `vector/geojson/streams.geojson` - Flüsse
- `vector/geojson/landtype.geojson` - Vegetationsformen

```
plot(dem)
plot(coastline, add = TRUE)
plot(streams, col = "#3b7fb2", add = TRUE)
plot(sites, pch = 16, add = TRUE)
```



Koordinatensystem und Projektion

```
streams
```

```
## class       : SpatialLinesDataFrame
## features    : 99
## extent      : -15.65803, -14.61021, 35.37663, 35.75371  (xmin, xmax, ymin, y
## coord. ref. : +proj=longlat +ellps=WGS84 +towgs84=0,0,0,0,0,0,0 +no_defs
## variables   : 6
## names       : PK_UID, cat, value, label, discharge, distance
## min values  :      1,   1,    10,    NA,         1,       NA
## max values  :     99,  99,    98,    NA,        20,       NA
```

Die meisten Funktionen benötigen ein projeziertes Koordinatensystem ==> UTM

```
epsg <- 32628
crs <- paste("+init=epsg:",epsg,"", sep="")
```

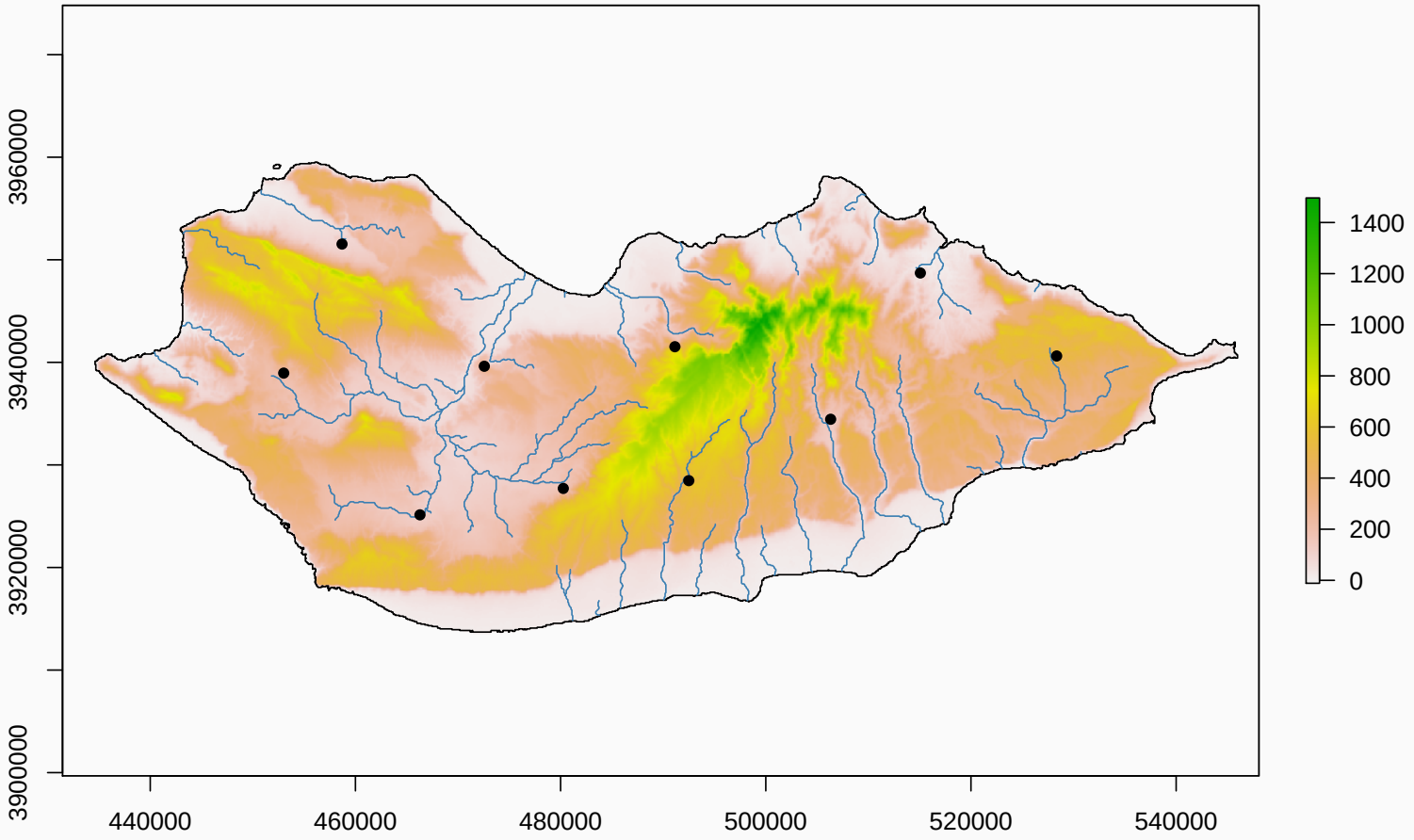
Projektion

```
dem <- projectRaster(dem, crs = crs )

coastline <-spTransform(coastline, CRS(crs))

landtype <- spTransform(landtype, CRS(crs))

sites <- spTransform(sites, CRS(crs))

streams <- spTransform(streams, CRS(crs))

streams
```

```
## class       : SpatialLinesDataFrame
## features    : 99
## extent      : 440403.1, 535307.3, 3914831, 3956769  (xmin, xmax, ymin, ymax)
## coord. ref. : +init=epsg:32628 +proj=utm +zone=28 +datum=WGS84 +units=m +no_
## variables   : 6
## names       : PK_UID, cat, value, label, discharge, distance
## min values  :      1,   1,    10,    NA,         1,       NA
## max values  :     99,  99,    98,    NA,        20,       NA
```

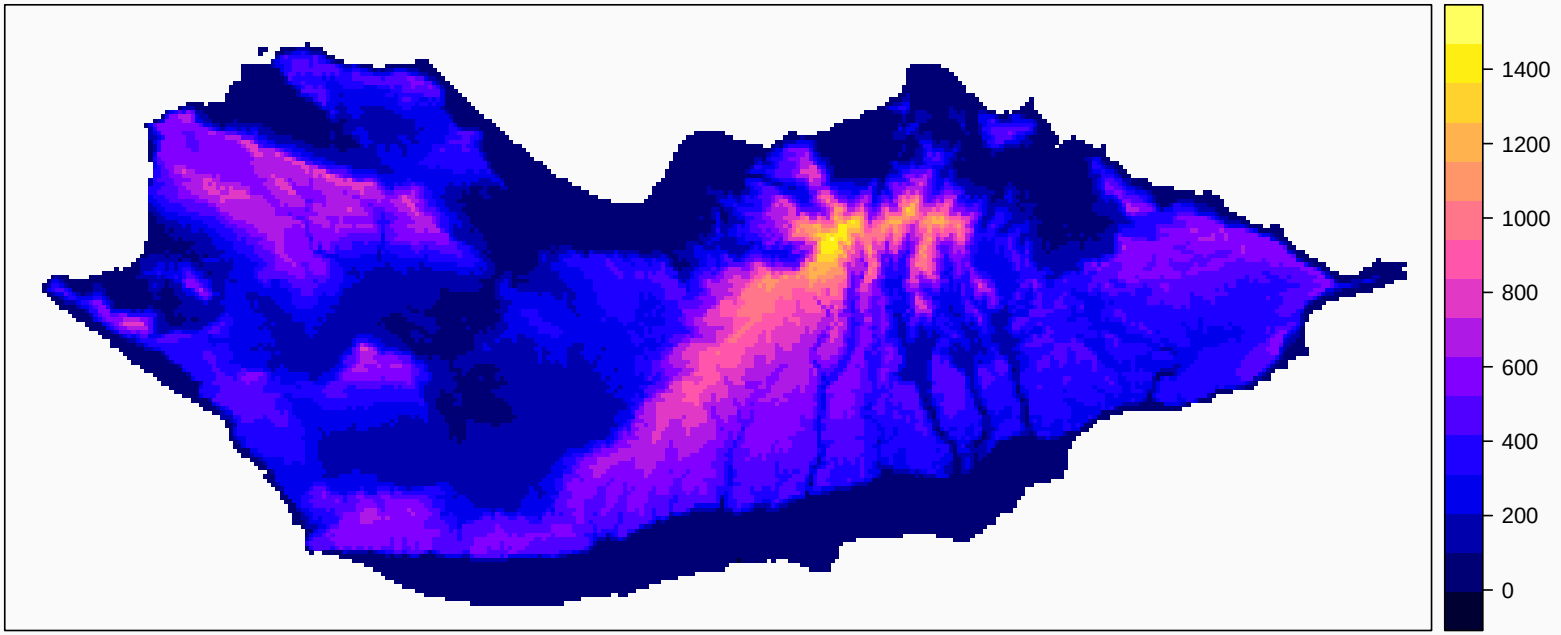
Projektion

```
plot(dem)
plot(coastline, add = TRUE)
plot(streams, col = "#3b7fb2", add = TRUE)
plot(sites, pch = 16, add = TRUE)
```



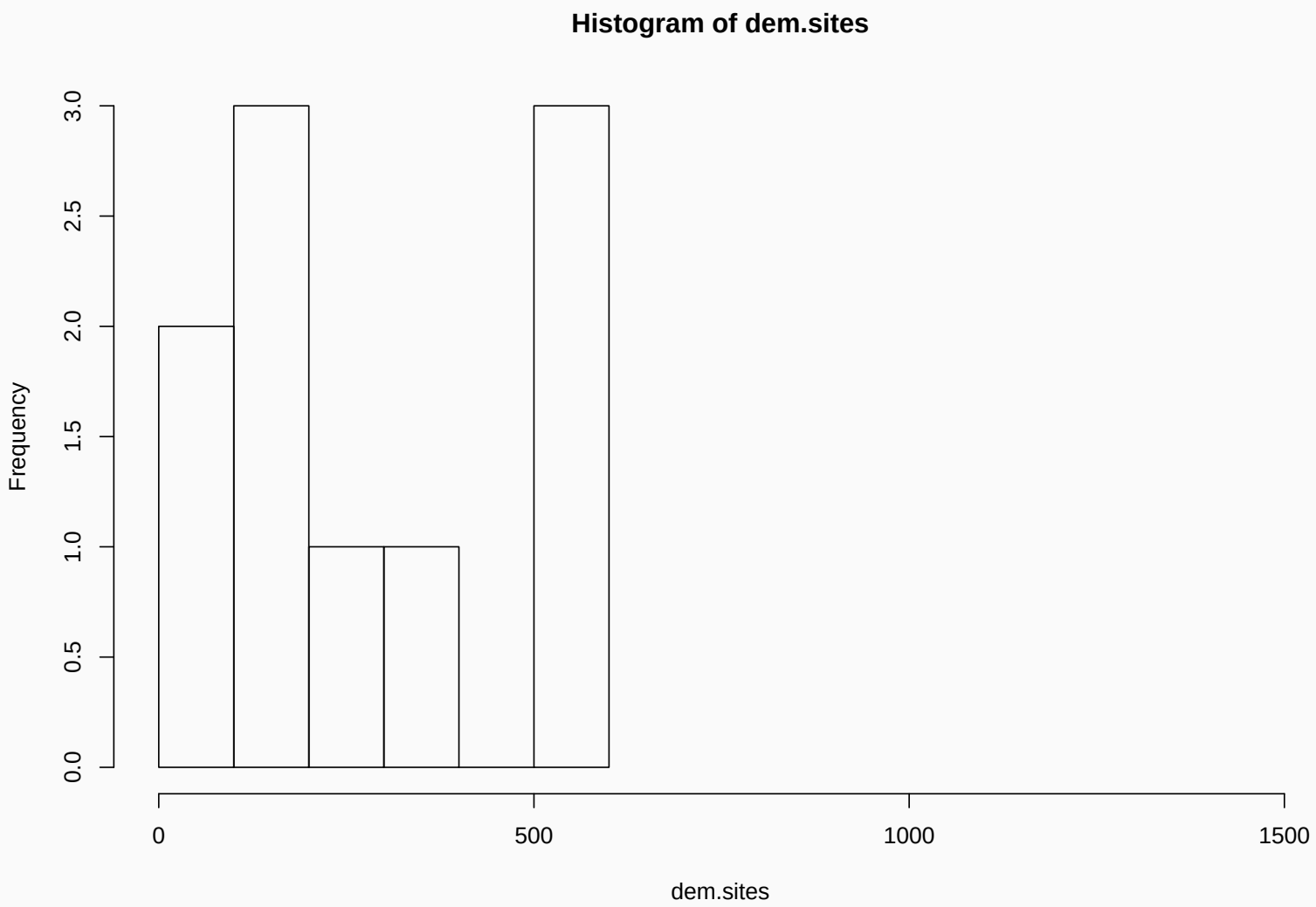
spplot()

```
spplot(dem)
```



Lage der Fundstellen

```
dem.sites <- extract(dem, sites)
dem.max <- max(getValues(dem), na.rm = TRUE) # höchster Punkt
hist(dem.sites, xlim = c(0, dem.max))
```



Lage der Fundstellen

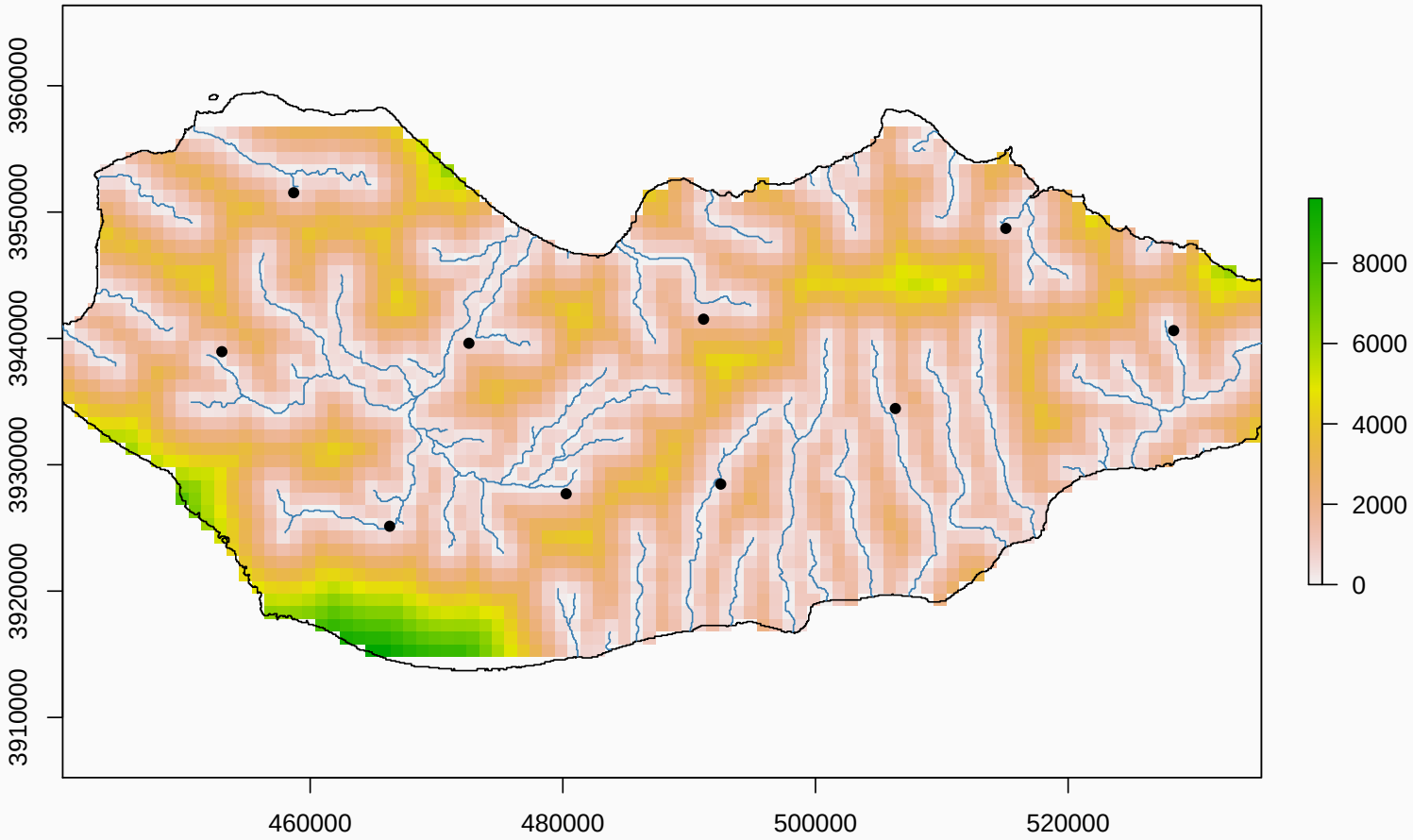
Entfernung zu den Flüssen

```
streams <- gUnion(streams, streams) # mögliche multi-polygone zusammenführen

ext <- extent(streams) # bounding box & resolution

ncol <- length((ext@xmin/1000):(ext@xmax/1000))
nrow <- length((ext@ymin/1000):(ext@ymax/1000))

streams.dist <- raster(extent(streams), nrow = nrow, ncol = ncol, crs = crs)

dd = gDistance(streams, as(streams.dist,"SpatialPoints"), byid=TRUE)

streams.dist[] = apply(dd,1,min)

streams.dist <- mask(streams.dist, coastline)
```

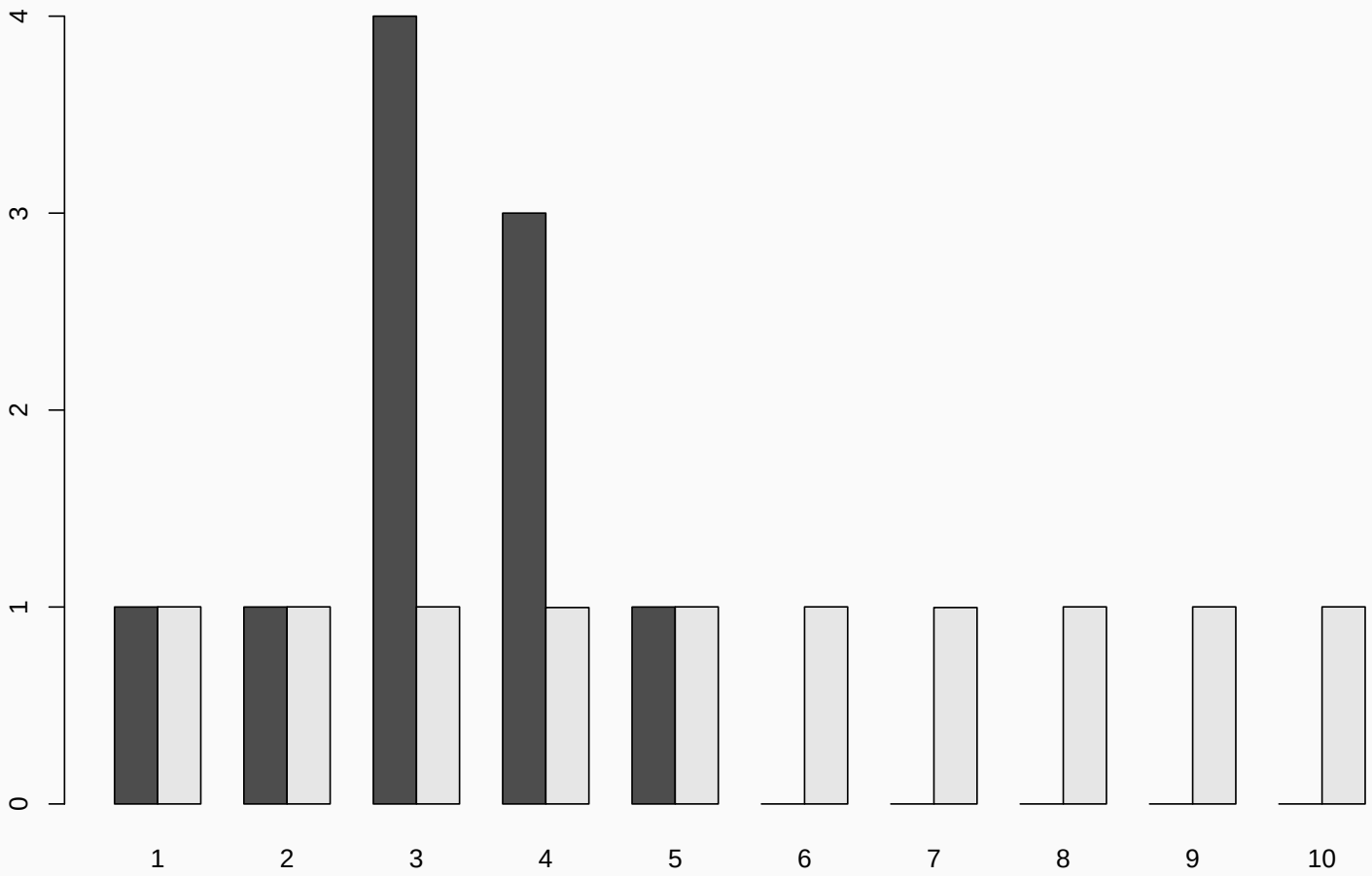
ext@xmin/1000 so gering, damit Berechnung schneller geht (hier ein 1km Raster)

Lage der Fundstellen

```
plot(streams.dist)
plot(coastline, add = TRUE)
plot(streams, col = "#3b7fb2", add = TRUE)
plot(sites, pch = 16, add = TRUE)
```

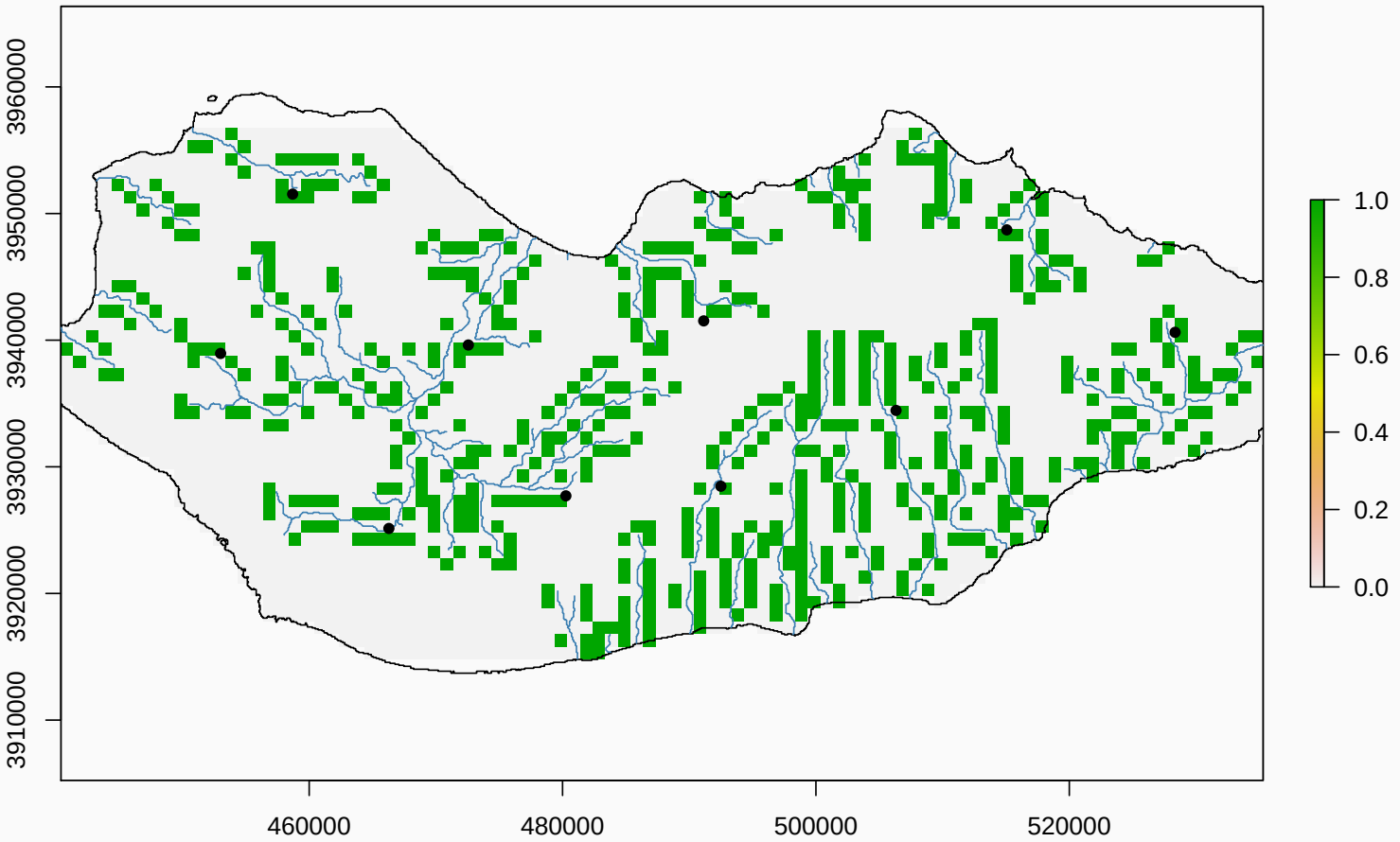


Lage der Fundstellen



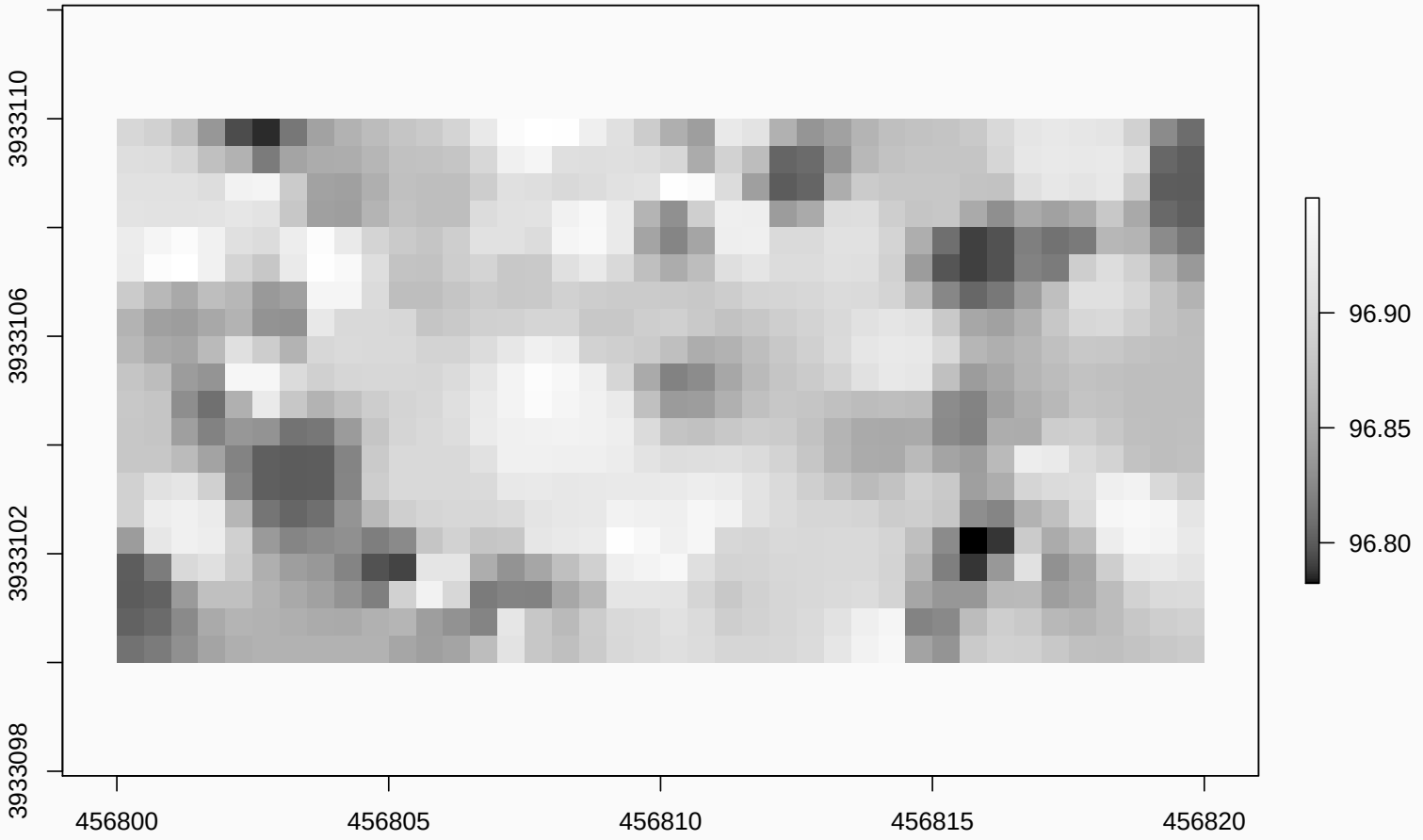

```
## [1] "Chi-Quadrat-Verteilungs-/Anpassungstest: p = 0.035037710884661"
```

Modelle


3960000
3950000
3940000
3930000
3920000
3910000
460000
480000
500000
520000
1.0
0.8
0.6
0.4
0.2
0.0

Geophysik

```
geophys_excavation <- raster("../../AtlantGIS-master/raster/geophys/geophys_exc
geophys_excavation <- projectRaster(geophys_excavation, crs = crs )

plot(geophys_excavation, col = gray.colors(256, start = 0, end = 1))
```

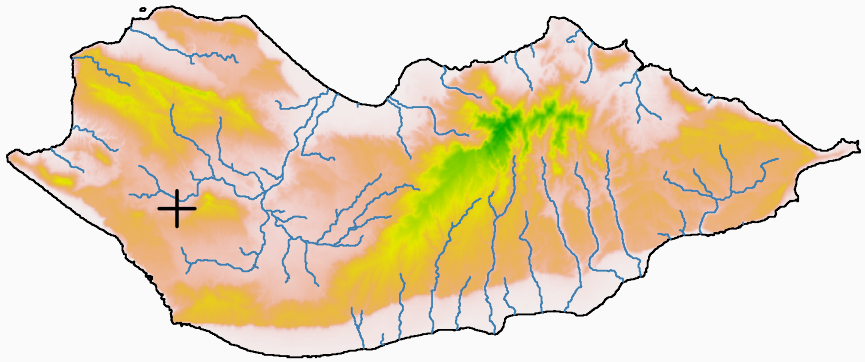


Geophysik

```
extent(geophys_excavation)
```

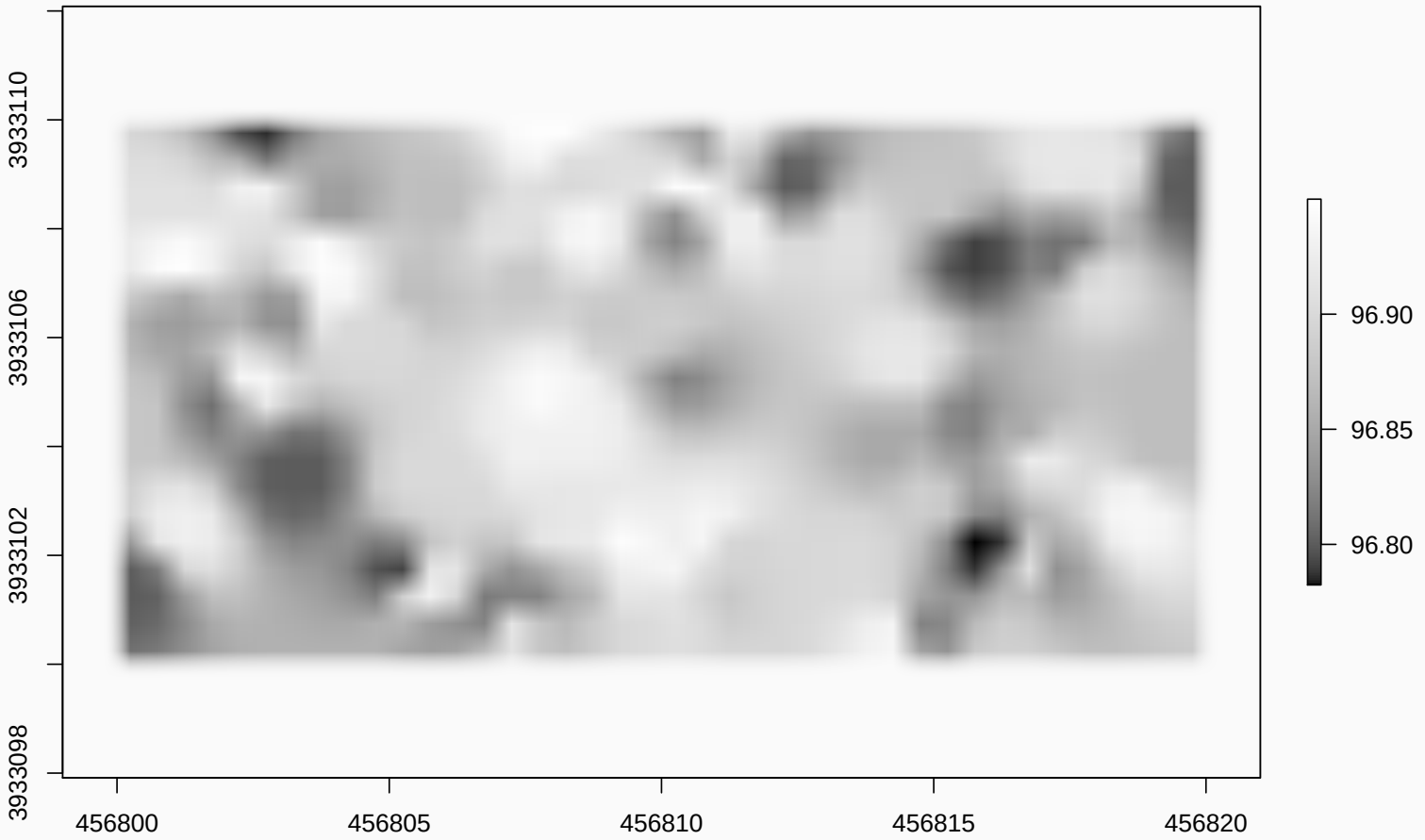
```
## class       : Extent
## xmin        : 456799
## xmax        : 456821
## ymin        : 3933099
## ymax        : 3933111
```

```
##    id    lat     lng
## 1 crt 456810 3933105
```



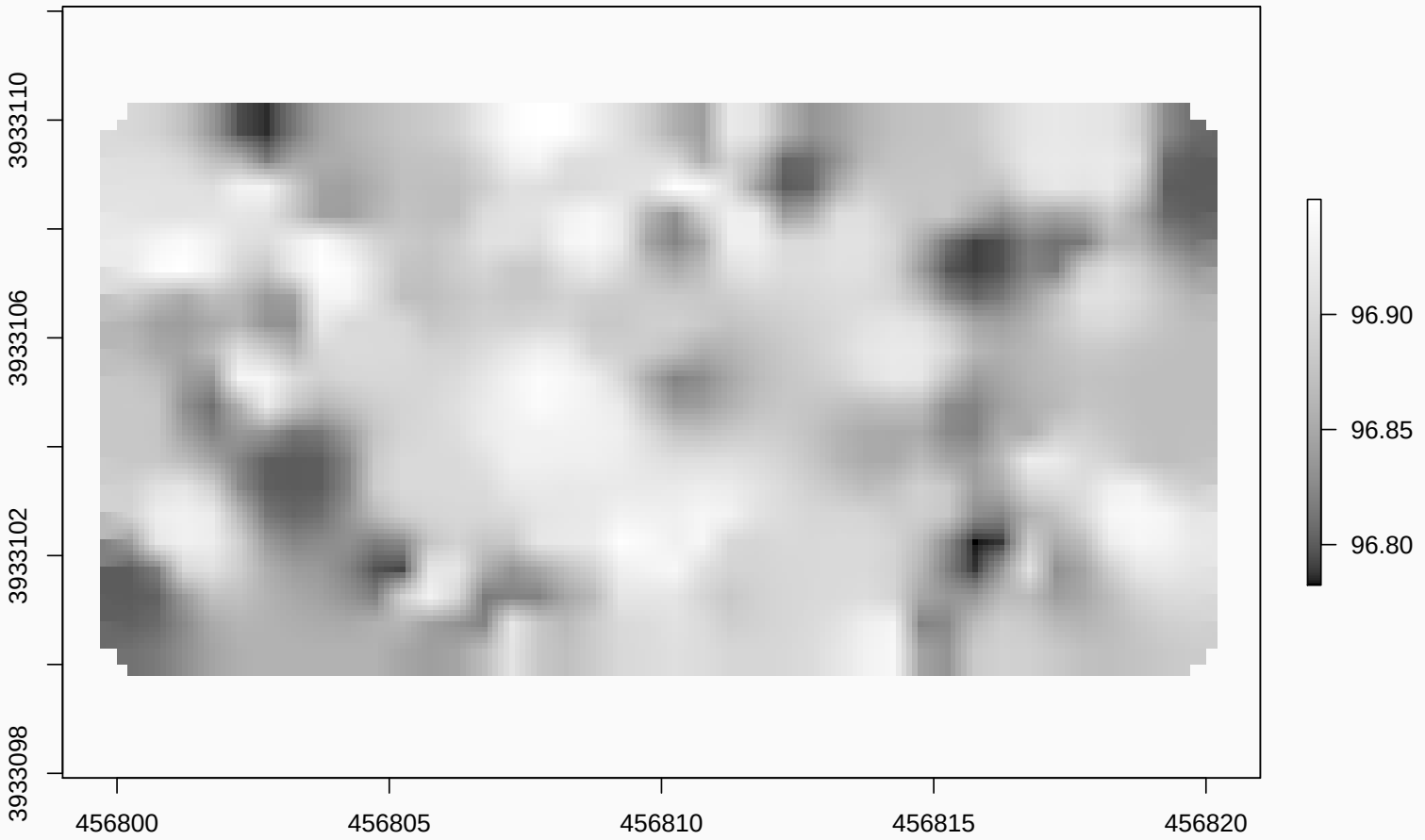
plot(..., interpolate=TRUE)

```
plot(geophys_excavation, interpolate=TRUE,
     col = gray.colors(256, start = 0, end = 1))
```



disaggregate()

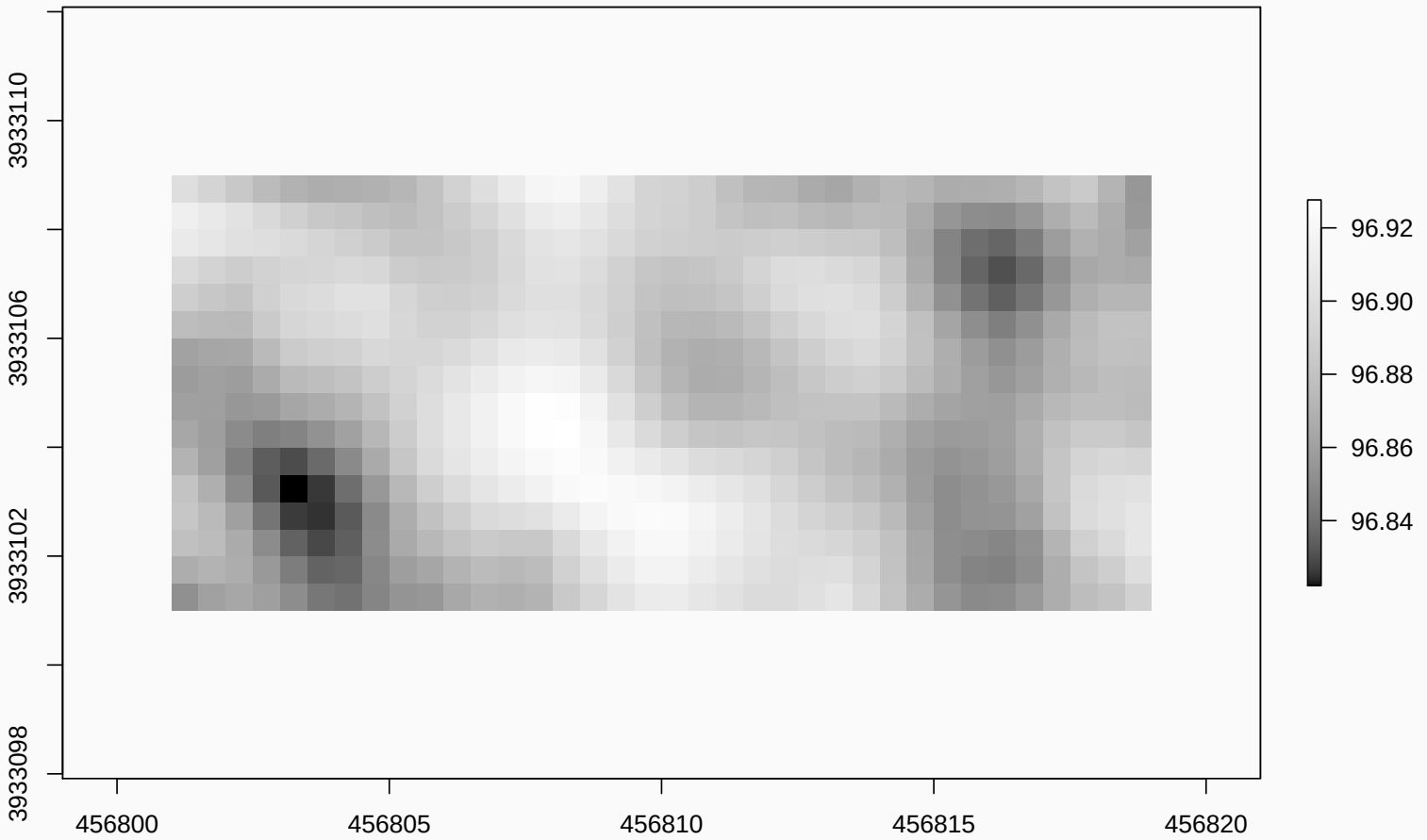
```
y <- disaggregate(geophys_excavation, 5, method='bilinear')
plot(y, col = gray.colors(256, start = 0, end = 1))
```



focal()

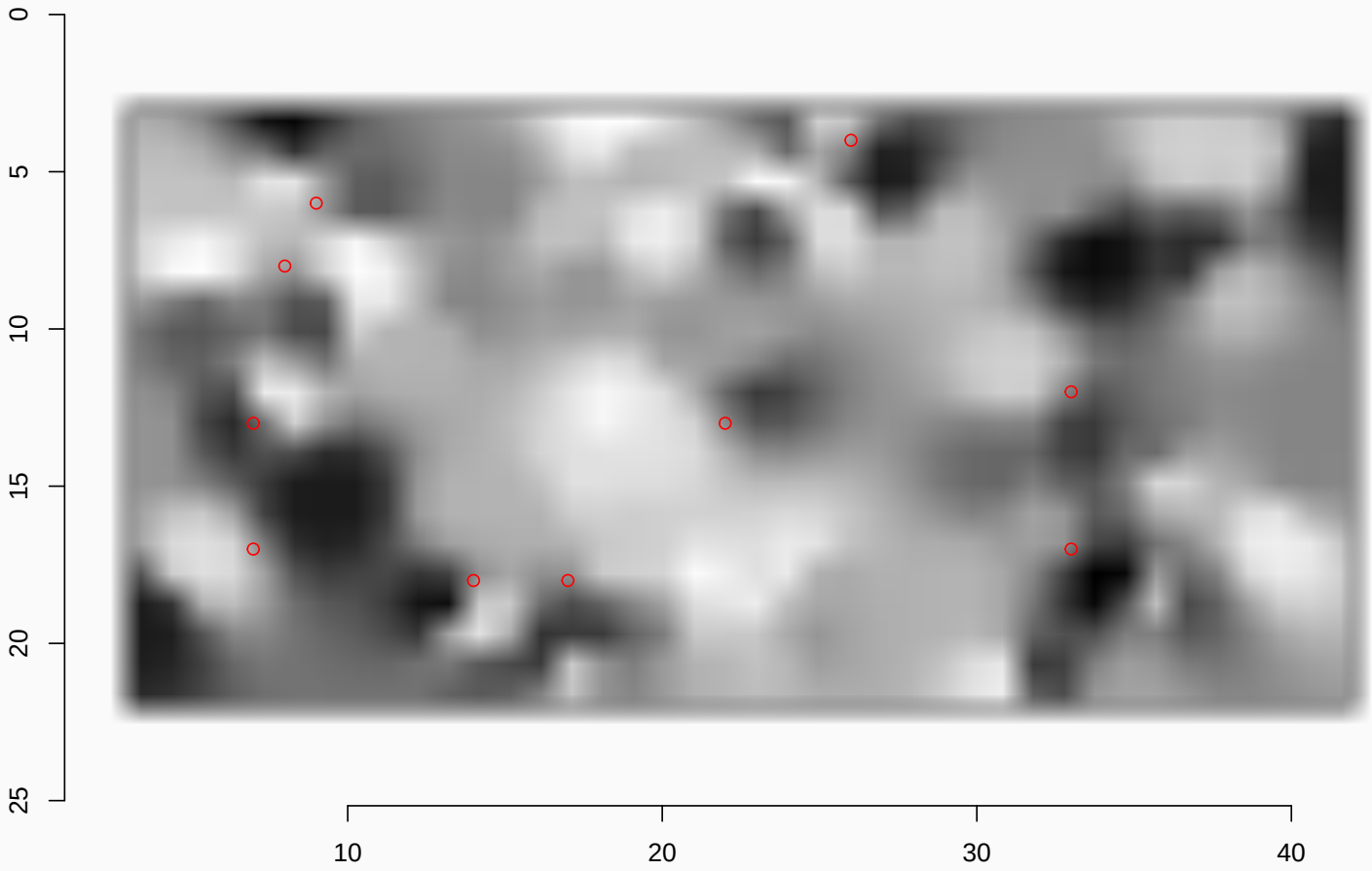
Berechnet für jede Eingabezellenposition eine Statistik der Werte innerhalb einer angegebenen Nachbarschaft

```
y <- focal(geophys_excavation, w=matrix(1, 5, 5), mean)
plot(y, col = gray.colors(256, start = 0, end = 1))
```



Anomalien finden

Blob detection/extraction of local maxima, denoising, scale-space



Weitere Daten

```
plot(geophys_excavation, col = gray.colors(256, start = 0, end = 1))
plot(features, add = TRUE)
plot(walls, col=rgb(0,100,0,50,maxColorValue=255), add = TRUE)
```