University of California, Los Angeles
Department of Statistics

Statistics C173/C273 **Instructor: Nicolas Christou**

Assign "NA" values outside the area defined by the observed data points

In this document we will discuss how to assign "NA" values to the area of the raster map outside the observed data points. We should always try to keep the predictions within the region defined by the data points. Kriging is an interpolator not extrapolator. We will use again the data from the Maas river at

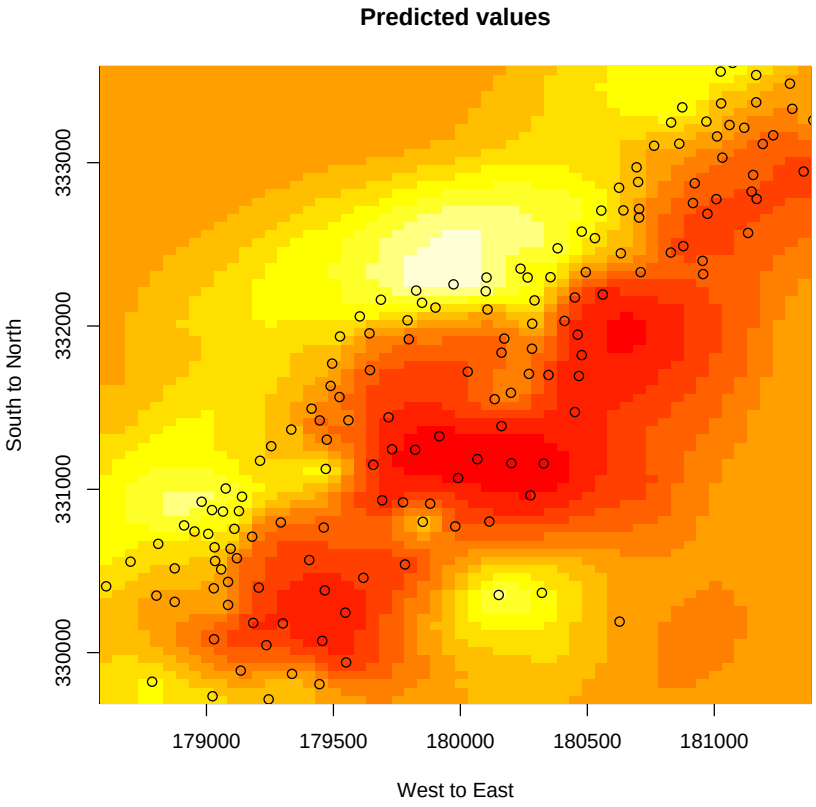
```
a <- read.table("http://www.stat.ucla.edu/~nchristo/statistics_c173_c273/
     soil.txt", header=T)
# Save the original image function:
image.orig <- image
```

We have also saved the original `image` function since we will work with the `gstat` package. The following commands have been discussed in previous documents as well.

```
#Create the grid:
x.range <- as.integer(range(a[,1]))
y.range <- as.integer(range(a[,2]))
grd <- expand.grid(x=seq(from=x.range[1], to=x.range[2], by=50),
y=seq(from=y.range[1], to=y.range[2], by=50))

#Create a gstat object:
g <- gstat(id="log_lead", formula = log(lead)~1, locations = ~x+y, data =a)
plot(variogram(g))

#Fit the variogram:
v.fit <- fit.variogram(variogram(g), vgm(0.5,"Sph",1000,0.1))
plot(variogram(g),v.fit)

#Perform ordinary kriging predictions:
pr_ok <- krige(id="log_lead",log(lead)~1, locations=~x+y,model=v.fit, data=a,
newdata=grd)

#Collapse the vector of the predicted values into a matrix:
qqq <- matrix(pr_ok$log_lead.pred, length(seq(from=x.range[1],
       to=x.range[2], by=50)), length(seq(from=y.range[1],
       to=y.range[2], by=50)))

#Create a raster map:
image.orig(seq(from=x.range[1], to=x.range[2], by=50),
seq(from=y.range[1],to=y.range[2],  by=50), qqq,
xlab="West to East",ylab="South to North", main="Predicted values")
```

```
#Add the observed data points on the raster map:
points(a)
```

The commands above will create the raster map:



We would like now to keep only the part of the raster map which is defined by the observed data points. We will use the following program which creates a window that scans the entire raster map area and assigns "NA" values to the coordinates beyond the observed data points.

```
#We want to assign "NA" values for the region outside the observed data
#points:
X <- seq(from=x.range[1], to=x.range[2], by=50)
Y <- seq(from=y.range[1], to=y.range[2], by=50)

mask <- function(predictions,pts,grid.x,grid.y,win.size) {
        new.qqq <- predictions
        m <- nrow(predictions)
        n <- ncol(predictions)
                  for (i in 1:m) {
                  for (j in 1:n) {
                  x.min <- i-win.size
                  y.min <- j-win.size
                  x.max <- i+win.size
                  y.max <- j+win.size
if (x.min < 1) { x.min <- 1 }; if (y.min < 1) { y.min <- 1 }
if (x.max > length(X)) { x.max <- length(X) };
if (y.max > length(Y)) { y.max <- length(Y) }

candidates <- pts[which(pts$x >= X[x.min] & pts$x <= X[x.max]),]

thePoints <- candidates[which(candidates$y >= Y[y.min] &
candidates$y <= Y[y.max]),]

if (nrow(thePoints) == 0) {
new.qqq[i,j] <- NA
                          }
                               }
                               }
return(new.qqq)
                                                          }
#Run the program:
d <- mask(qqq,a,X,Y,5)
```

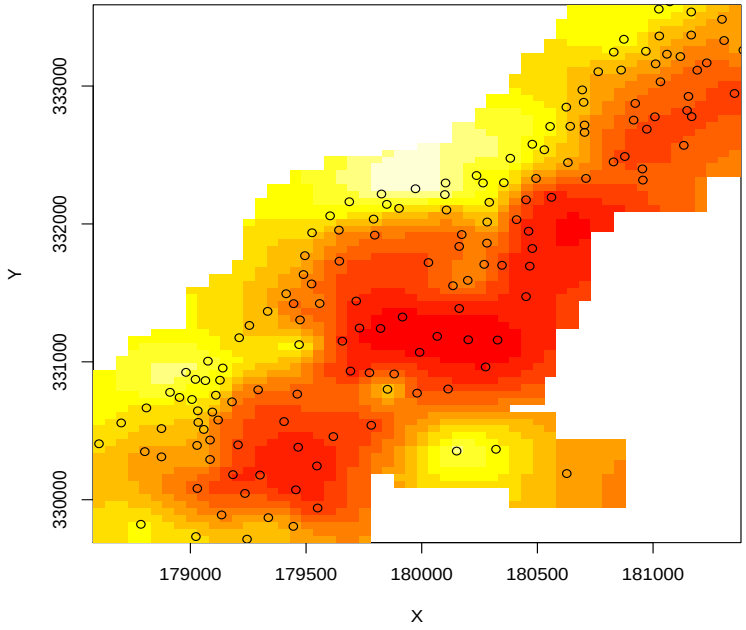
We can now create the new raster map:

```
#Create the new raster map:
image.orig(X,Y,d)
points(a)
```

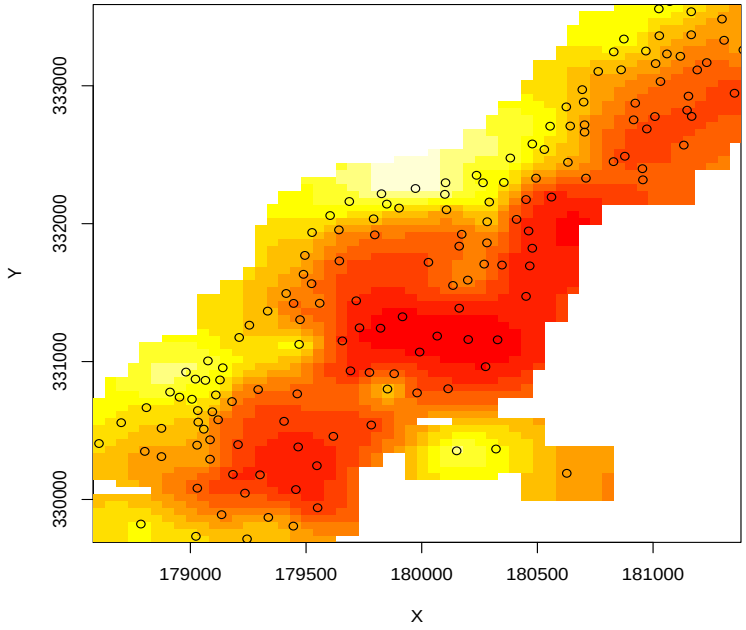
In case we wanted to change the values of the axes we do the following:

```
image.orig(X,Y,d, axes=FALSE)
x.at <- seq(179000, 181000, by=100)
y.at <- seq(330000, 333000, by=100)
axis(1, at=x.at)
axis(2, at=y.at)
```

Here is the new raster map (size of window is 5):



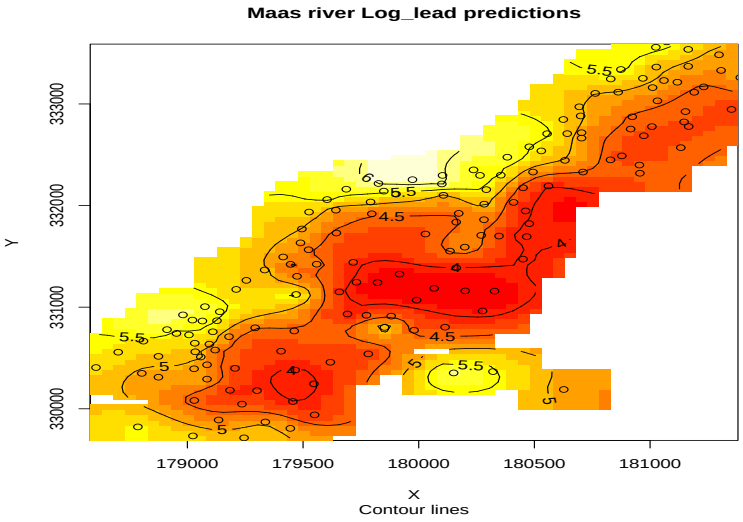
And here is the new raster map with window size 4:



Add contour lines to the raster map:
We can add contours lines to the raster map as follows:

```
#Add contour lines:
contour(seq(from=x.range[1], to=x.range[2], by=50),
seq(from=y.range[1],to=y.range[2], by=50), d,
levels=seq(1, 6, by=.5), add=TRUE, col="black", labcex=1)

title(main="Maas river Log_lead predictions", sub = "Contour lines")
```

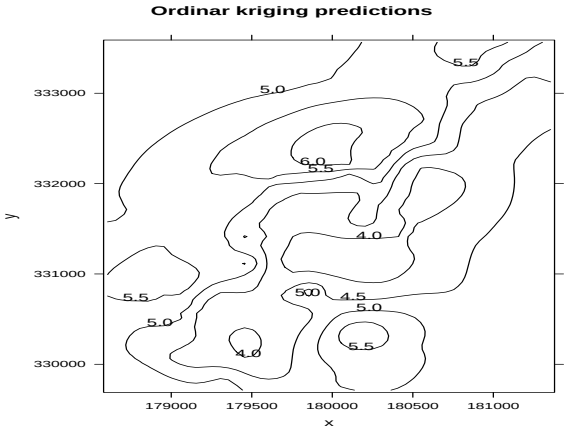
Here is the plot:



We can also create a contour map using the `contourplot` function:

```
#Another way to create contour lines:
contourplot(pr_ok$log_lead.pred~x+y,pr_ok,aspect ="iso",
main="Ordinary kriging predictions")
```

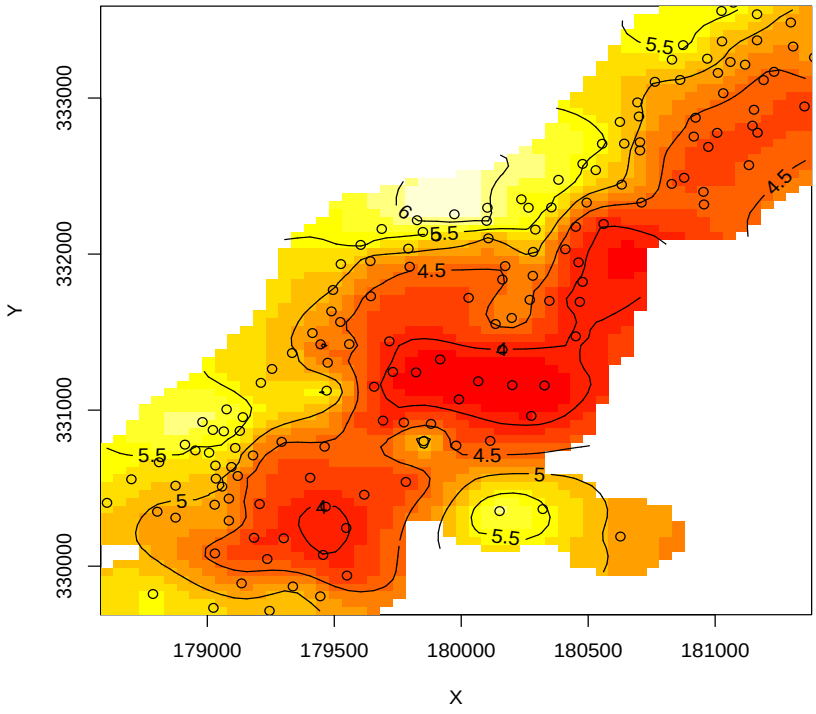
And the plot:



Another function:

```
X <- seq(from=x.range[1], to=x.range[2], by=50)
Y <- seq(from=y.range[1], to=y.range[2], by=50)

dz.mask <- function(grid, pts, x.dom, y.dom, window, mitre=2) {
N <- length(pts[ ,1]) ; mask <- array( NA, dim(grid) )
for(j in 1:N) {
dx <- abs(x.dom - pts$x[j])
dy <- abs(y.dom - pts$y[j])
d.Mx <- tcrossprod( dx , rep(1, length(dy)) )^mitre +
tcrossprod( rep(1, length(dx)), dy )^mitre
mask[ d.Mx < window^mitre ] <- FALSE
}
return(mask+grid)
}

qqq.masked <- dz.mask(qqq, a, X, Y, 250)

image.orig(X,Y,qqq.masked)
points(a)

contour(seq(from=x.range[1], to=x.range[2], by=50),
seq(from=y.range[1],to=y.range[2], by=50), d,
levels=seq(1, 6, by=.5), add=TRUE, col="black", labcex=1)
```

And the raster map:



Assign NA values outside maps of the package "maps"

Consider the California ozone data:

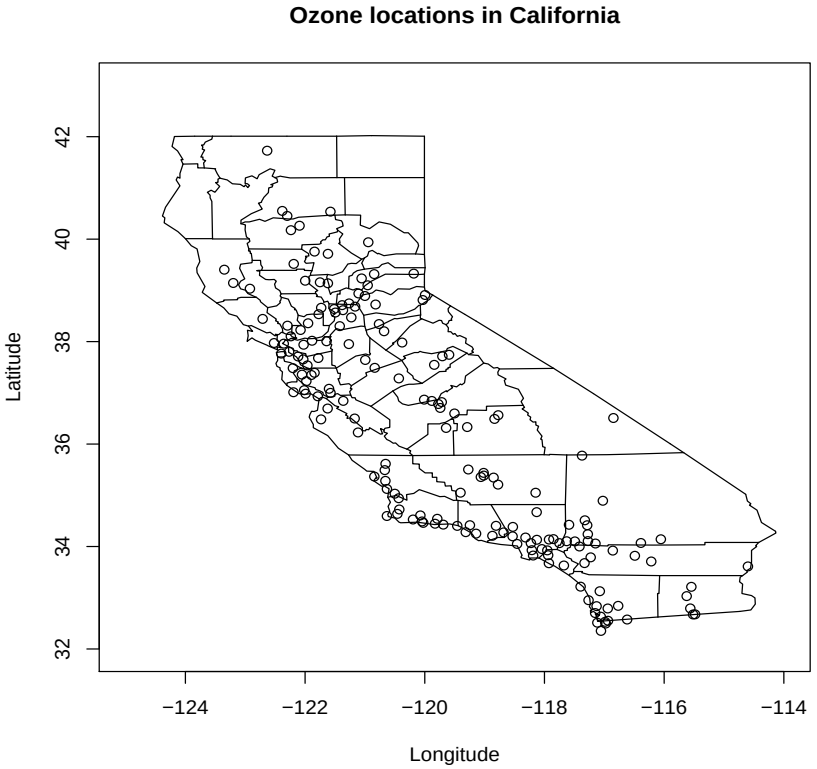
```
a<-read.table("http://www.stat.ucla.edu/~nchristo/statistics_c173_c273/
   ozone.txt", header=T)
```

First save the original image function `image.orig <- image`.
We will use the following packages:

```
library(gstat)
library(maps)
```

A simple plot of the data points on the map:

```
plot(a$x,a$y, xlim=c(-125,-114),ylim=c(32,43), xlab="Longitude",
ylab="Latitude", main="Ozone locations in California")

map("county", "ca",add=TRUE)
```

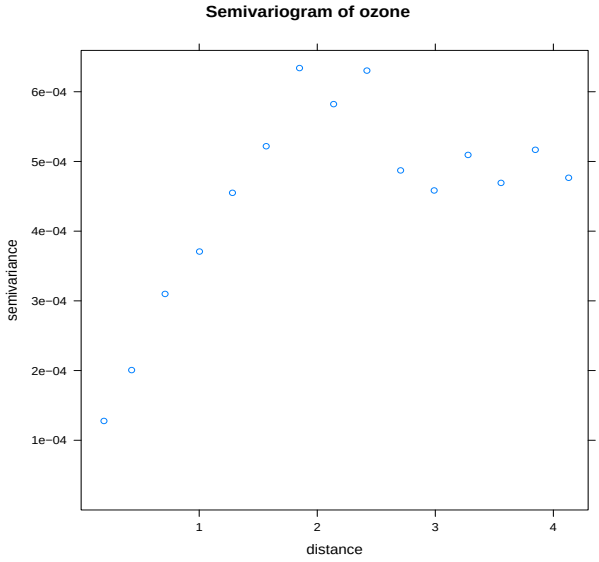


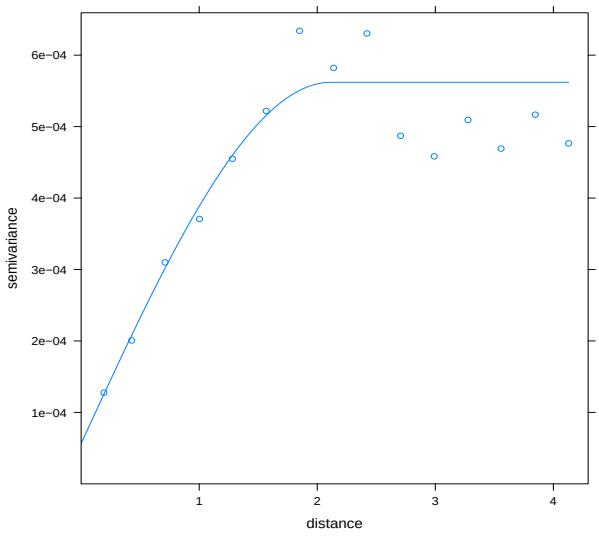
The following commands will create a `gstat` object, compute the sample variogram, and fit a model to the sample variogram.

Here are the commands:

```
g <- gstat(id="o3", formula = o3~1, locations = ~x+y,
data = a)

plot(variogram(g), main="Semivariogram of ozone")

v.fit <- fit.variogram(variogram(g), vgm(0.0006,"Sph",3,0.0001))

plot(variogram(g),v.fit)
```

And here are the plots:





Create the grid for kriging predictions:

```
x.range <- c(-125,-114)
y.range <- c(32, 43)

grd <- expand.grid(x=seq(from=x.range[1], to=x.range[2], by=0.1),
y=seq(from=y.range[1], to=y.range[2], by=0.1))
```

Perform ordinary kriging predictions:

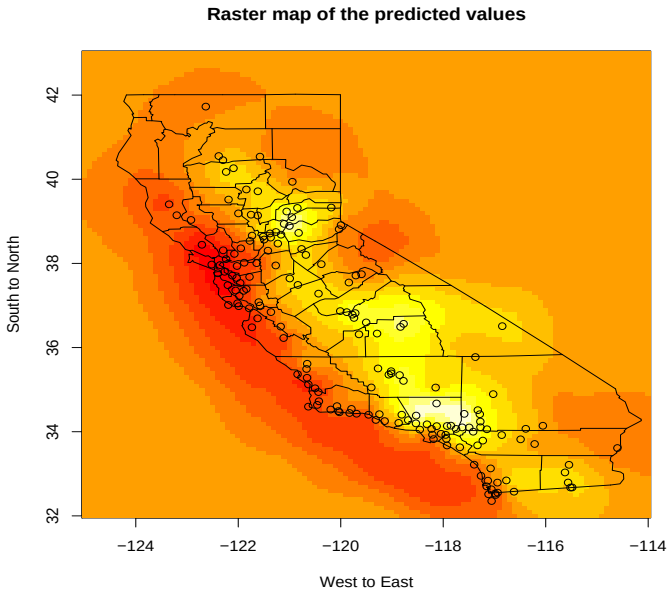
```
pr_ok <- krige(id="o3",o3~1, locations=~x+y, model=v.fit, data=a,
         newdata=grd)
```

Collapse the vector of the predicted values into a matrix:

```
qqq <- matrix(pr_ok$o3.pred, length(seq(from=x.range[1],
       to=x.range[2], by=0.1)), length(seq(from=y.range[1],
       to=y.range[2], by=0.1)))
```

The following commands will construct the raster map of the predicted values, place the data points on the map, and add the map of California.

```
image.orig(seq(from=x.range[1],to=x.range[2],by=0.1),
seq(from=y.range[1],to=y.range[2], by=0.1),qqq,
xlab="West to East",ylab="South to North", main="Predicted values")

points(a)

map("county", "ca",add=TRUE)
```



We want now to assign "NA" values to the grid values outside the boundaries of the state of California. This can be done easily with the `map.where` function as follows:

```
#Identify the location of each point in the grid:
in.what.state <- map.where(database="state", x=grd$x, y=grd$y)

#Find the points of the grid that belong to California:
in.ca <- which(in.what.state=="california")

pred <- pr_ok$o3.pred

#Assign NA values to all the points outside California:
pred[-in.ca] <- NA
```

The vector `pred` contains the California points plus the NA values outside California. This vector will be collapsed into a matrix below:

```
qqq.ca <- matrix(pred, length(seq(from=x.range[1], to=x.range[2], by=0.1)),
length(seq(from=y.range[1], to=y.range[2], by=0.1)))
```

We can now construct the new raster map:

```
image.orig(seq(from=x.range[1],to=x.range[2],by=0.1),
seq(from=y.range[1],to=y.range[2], by=0.1),qqq.ca,
xlab="West to East",ylab="South to North", main="Predicted values")
```

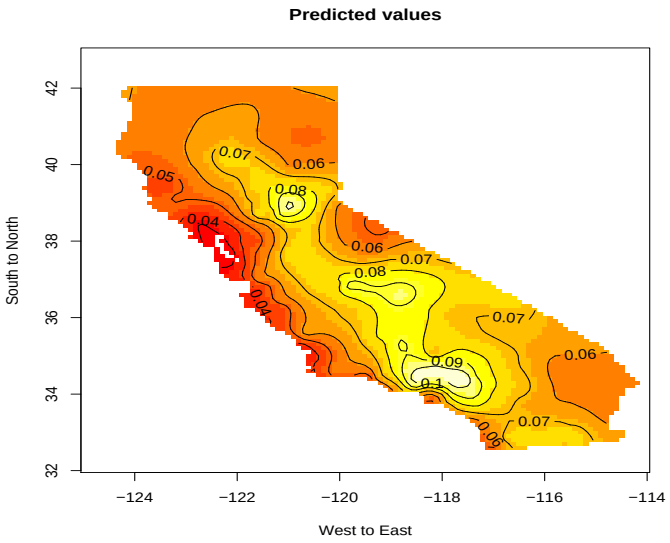
And we can add contours:

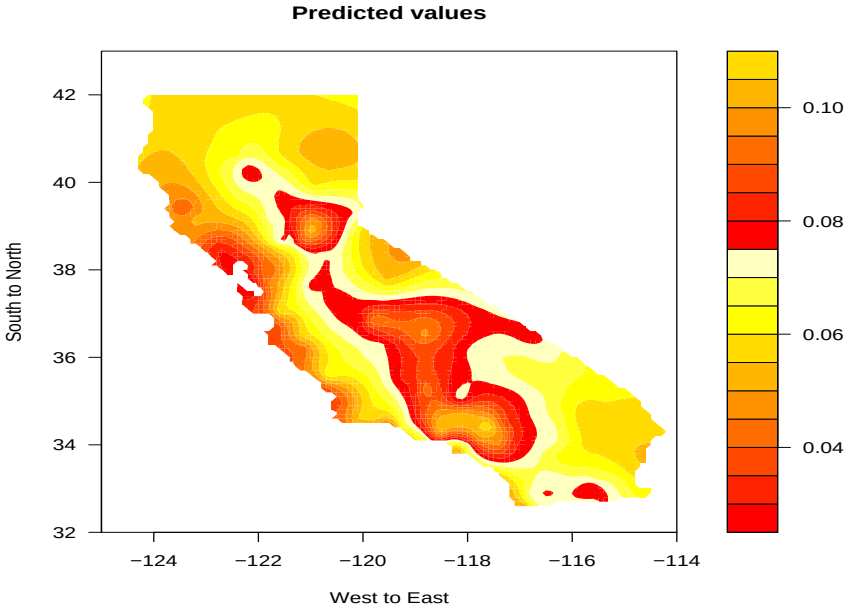
```
contour(seq(from=x.range[1], to=x.range[2], by=0.1),
seq(from=y.range[1],to=y.range[2], by=0.1), qqq.ca, add=TRUE,
col="black", labcex=1)
```

Another function for contours is the following:

```
filled.contour(seq(from=x.range[1],to=x.range[2],by=0.1),
seq(from=y.range[1],to=y.range[2], by=0.1),qqq.ca,
xlab="West to East",ylab="South to North", main="Predicted values",
col=heat.colors(10))
```

The two plots are shown on the next page:


Predicted values
South to North
West to East
42
40
38
36
34
32
−124
−122
−120
−118
−116
−114
0.07
0.06
0.05
0.08
0.04
0.09
0.1


Predicted values
South to North
West to East
42
40
38
36
34
32
−124
−122
−120
−118
−116
−114
0.10
0.08
0.06
0.04