SISMID 2024 Spatial Statistics Waller Point Process 2: Dental Intensities

Lance A. Waller

7/11/2024

Spatial point processes: Intensities and archaeology

Grave dental data from Richard Wright in Australia, Emeritus Professor, School of Archaeology, University of Sydney NSW 2006, Australia

We'll need two libraries for this... there are fancier options now, but these will do the trick.

```
library(splancs)
```

```
## Loading required package: sp
```

```
##
## Spatial Point Pattern Analysis Code in S-Plus
##
##  Version 2 - Spatial and Space-Time analysis
```

```
library(KernSmooth) # Matt Wand's kernel smoothing techniques
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
library(here)
```

```
## here() starts at /Users/lwaller/Library/CloudStorage/OneDrive-Emory/meetings/SISMID.2024/SISMID_2024_
```

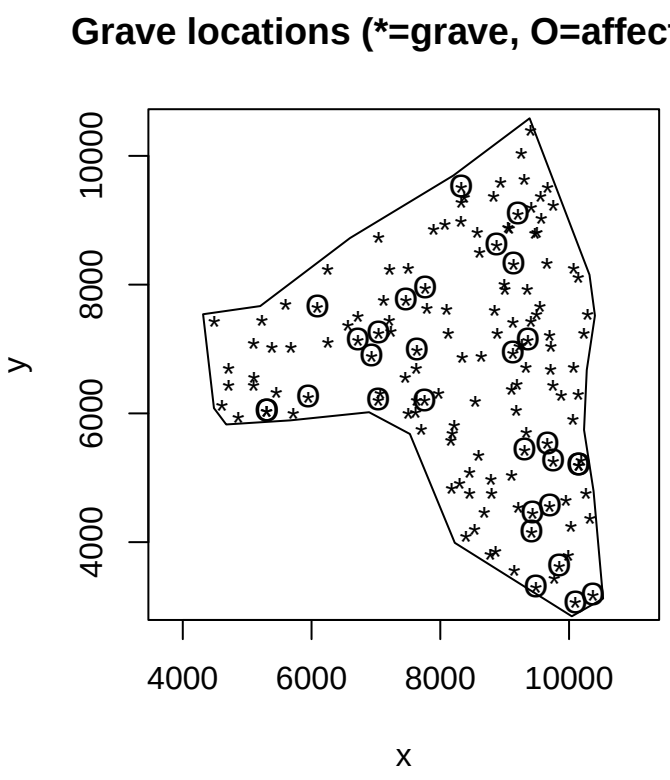
Set working directory and read in the data (dental.reduced.dat) and a polygon outline file (created earlier... commented code below shows how to create your own in splancs).

```
######################
# Set path to data sets, etc.
######################

# Set my working directory (Lance's here for example)
#setwd("~/OneDrive - Emory University/meetings/SISMID.2021/SISMID.2021.Waller.Rcode")

# Read in data with south region removed
dental <- scan(here("data", "dental.reduced.dat"),list(lab=0,aff=0,x=0,y=0))

# read in previously defined polygon boundary
mypoly <- read.table(here("data", "mypoly.dat"))
```

Now to set up the data and make preliminary plots (with some adjustments to have square plots).

```
# Set points as spatial point process data for splancs
dental.p <- as.points(dental$x,dental$y)
dentcas.p <- as.points(dental$x[dental$aff==1],dental$y[dental$aff==1])
```

```r
dentcon.p <- as.points(dental$x[dental$aff==0],dental$y[dental$aff==0])

par(pty="s")

# dental$y has a bigger range so this tries to add half of the
# extra to each side

extra <- ( diff(range(dental$y))-diff(range(dental$x)) )/2

plot(dental.p,pch="*",xlim=c(min(dental$x)-extra,max(dental$x)+extra),
     ylim=range(dental$y),
     xlab="x",ylab="y")
title("Grave locations (*=grave, O=affected)")

points(dentcas.p,pch="O")

## Interactively define polygon -- run this function and click
# in the plot where you want the vertices of the polygon
# to be.  ("Connect the dots" around the points, then
# click right button to select 'done').
# mypoly <- getpoly()

# Add bounding polygon to the plot.
polygon(mypoly)
```



The next section considers how to *compare* two intensity functions. Do the peaks and valleys line up?

In spatial epidemiology we are interested in comparing the intensity of cases to the intensity of controls.

Where do these differ? Where are cases more likely than controls, or vice versa?

Since each spatial intensity (the expected number of events at a certain location) is proportional to the spatial density (the probability of observing an event at a certain location), if we take the ratio of to intensities, this corresponds to the *spatial relative risk* of observing a case event versus a control event. Mapping the spatial relative risk function gives a sense of where cases are more likely than controls. (Kelsall and Diggle, 1995 *Statistics in Medicine*).

The "sparr" package does a lot of this automatically, but to see how it works, here is the "classic" code used to generate this for the Waller and Gotway textbook.

First, we get the case intensity and the control intensity *with the same bandwidth* (so the two surfaces have the same flexibility and we're not finding noise over signal). The code below uses the "KernSmooth" package to get kernel smoothed density estimates (proportional to intensity estimates).

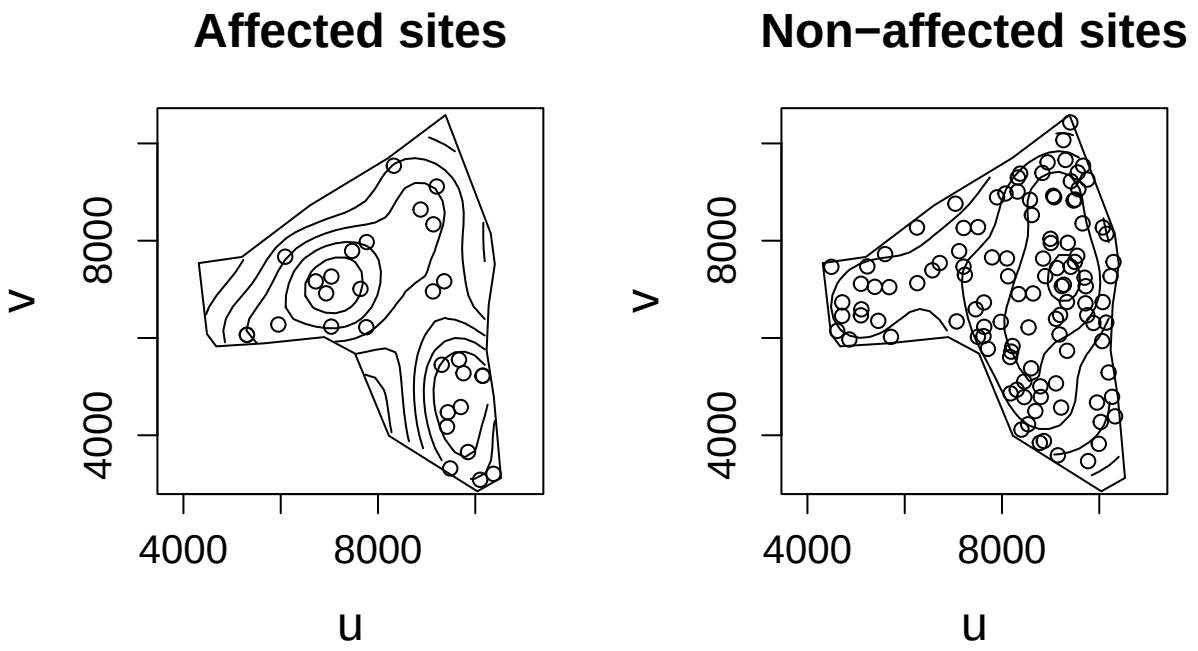
We will define the grid of points at which we calculate the estimate

We're setting the bandwidth to 700 distance units. Calculating the density estimates and plotting both.

```
##############################################################
# Kelsall and Diggle for bandwidth = 700
#pdf("dental.intensities.700.pdf",width=11,height=8)
par(mfrow=c(1,2),pty="s")

casbandw <- 700
conbandw <- 700

# define number of grid points in a row and in a column (same number for now)
grid.number <- 40

casdens <- bkde2D(dentcas.p, casbandw, gridsize=c(grid.number,grid.number),
                  range.x=list( c( min(dentcas.p[,1],dentcon.p[,1])-1.5*casbandw,
                                   max(dentcas.p[,1],dentcon.p[,1])+1.5*casbandw),
                                c( min(dentcas.p[,2],dentcon.p[,2])-1.5*casbandw,
                                   max(dentcas.p[,2],dentcon.p[,2])+1.5*casbandw) )     )

condens <- bkde2D(dentcon.p, conbandw, gridsize=c(grid.number,grid.number),
                  range.x=list( c( min(dentcas.p[,1],dentcon.p[,1])-1.5*casbandw,
                                   max(dentcas.p[,1],dentcon.p[,1])+1.5*casbandw),
                                c( min(dentcas.p[,2],dentcon.p[,2])-1.5*casbandw,
                                   max(dentcas.p[,2],dentcon.p[,2])+1.5*casbandw) )     )

# use splancs to identify points in mypoly polygon
# and set density estimates to NA outside of bounding polygon

gridlocs.y <- rep(casdens$x2[1],grid.number)
for (i in 2:grid.number) {
   gridlocs.y <- c(gridlocs.y,rep(casdens$x2[i],grid.number))
}
gridlocs.x <- rep(casdens$x1,grid.number)

gridlocs <- as.points(gridlocs.x,gridlocs.y)
inside <- inout(gridlocs,mypoly)
```

```
## Warning in as.points(poly): List/data.frame components should be named x and y
```

```
# This will set any grid point that is "not inside" (!inside) to NA so it won't show up
# in the plot.
casdens$fhat[!inside] <- NA
condens$fhat[!inside] <- NA

# 'persp' will make a fishnet plot of the surface
#persp(casdens$x1,casdens$x2,casdens$fhat,theta=30,phi=45,xlab="u",
#      ylab="v", zlab="Intensity",cex=2)
#title("Estimated intensity function",cex.main=1.75)

# 'plot' followed by 'contour' will plot the dots, then add (add=T) the contours to the plot
plot(dentcas.p,xlim=c(min(dental$x)-extra,max(dental$x)+extra),
     ylim=range(dental$y),
     xlab="u",ylab="v",cex.axis=1.25,cex.lab=1.5)
polygon(mypoly)
title("Affected sites",cex.main=1.5)
contour(casdens$x1,casdens$x2,casdens$fhat,add=T,
        vfont=c("sans serif","bold"),
        levels=c(0,0.00000001,0.00000002,0.00000003,0.00000004,0.00000005),
        drawlabels=F)
# if you want to add labels to the contours, replace the last ')' with a comma
#        labels=c(0,0.00000001,0.00000002,0.00000003,0.00000004,0.00000005),
#        labcex=0.8)

# now for the control surface
#persp(condens$x1,condens$x2,condens$fhat,theta=30,phi=45,xlab="u",
#      ylab="v", zlab="Intensity",cex=2)
#title("Estimated intensity function",cex.main=1.75)

plot(dentcon.p,xlim=c(min(dental$x)-extra,max(dental$x)+extra),
     ylim=range(dental$y),
     xlab="u",ylab="v",cex.axis=1.25,cex.lab=1.5)
polygon(mypoly)
title("Non-affected sites",cex.main=1.5)
contour(condens$x1,condens$x2,condens$fhat,add=T,
        vfont=c("sans serif","bold"),
        levels=c(0,0.00000001,0.00000002,0.00000003,0.00000004,0.00000005),
        drawlabels=F)
```




```
#         labels=c(0,0.00000001,0.00000002,0.00000003,0.00000004,0.00000005),
#         labcex=0.8)
#dev.off()
```

Now we'll calculate the relative risk surface by finding the ratio of the density estimates at the same grid points.

```
############################################################
# Kelsall and Diggle's ratio of intensity surfaces using KernSmooth

# Define the relative risk surface as the
# point-by-point ratio of case density to control density
r.ks <- log(casdens$fhat/condens$fhat)
# set any 'divide by zero' problems to 'NA'
r.ks[!is.finite(r.ks)] <- NA

# leave out the grid points outside the polygon
r.ks[!inside] <- NA
```

We'll be doing a Monte Carlo test at each grid point.

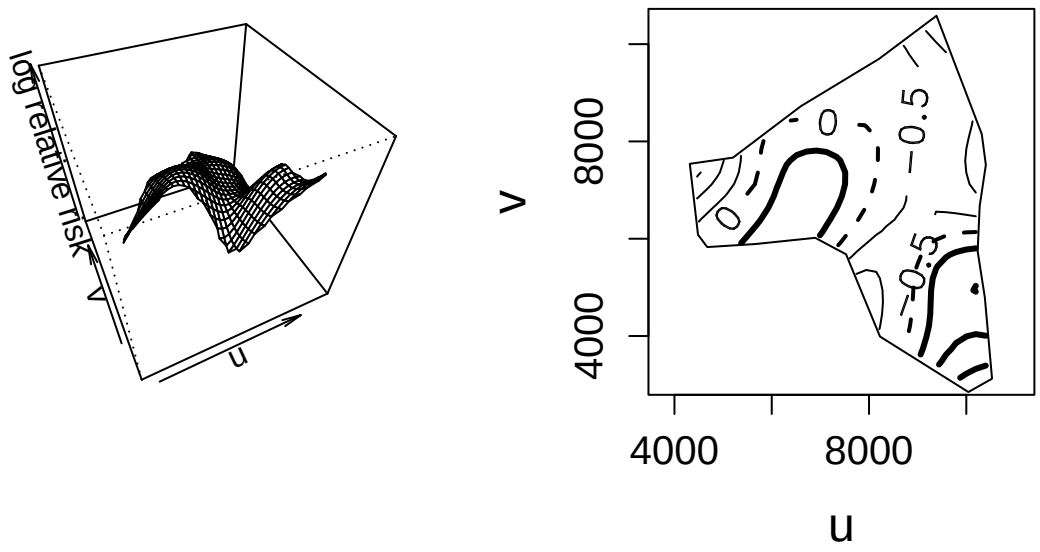
So we need a matrix of all of the test statistics (one for each grid point).

```
# calculate overall test statistic (divide each value of r.ks by area
# of grid squares

test.stat.obs <- sum( (r.ks[!is.na(r.ks)]/(diff(casdens$x1)[1]*diff(casdens$x2)[1]))^2 )
```

Now to plot the relative risk surface.

```
################################
# Plot relative risk surface

#pdf(paste("dentalKD.tol700.KS.pdf",width=11,height=8)
    par(mfrow=c(1,2),pty="s")
```

```
    persp(casdens$x1,casdens$x2,r.ks,theta=-25,phi=45,
          xlab="u",ylab="v",zlab="log relative risk",zlim=c(-3,3))

    title("Log relative risk surface")

    # dental$y has a bigger range so this tries to add half of the
    # extra to each side
    extra <- ( diff(range(dental$y))-diff(range(dental$x)) )/2

    plot(dentcon.p,pch=" ",xlim=c(min(dental$x)-extra,max(dental$x)+extra),
        ylim=range(dental$y),xlab="u",ylab="v",cex.axis=1.25,cex.lab=1.5)
    contour(casdens$x1,casdens$x2,r.ks,levels=c(-2,-1.5,-1,-0.5,0,0.5,1,1.5,2),
        lwd=c(1,1,1,1,2,3,3,3,3),
        lty=c(1,1,1,1,2,1,1,1,1),
        vfont=c("sans serif","bold"),
        labcex=1.25,add=T )
    polygon(mypoly)

# you can add the points if you want to.  'pch' = 'plot character'.
# pch = 16 is a filled circle
# pch = 1 is an open cirle
#   points(dentcas.p,pch=16)
#   points(dentcon.p,pch=1)

#Add a title pasting the bandwidth
   title(paste("Bandwidth =",casbandw))
```




```
#dev.off()
```

Now we want to figure out where the relative risk surface is more (or less) that we would expect under the null hypothesis where the local relative risk is 1.0 (the null hypothesis).

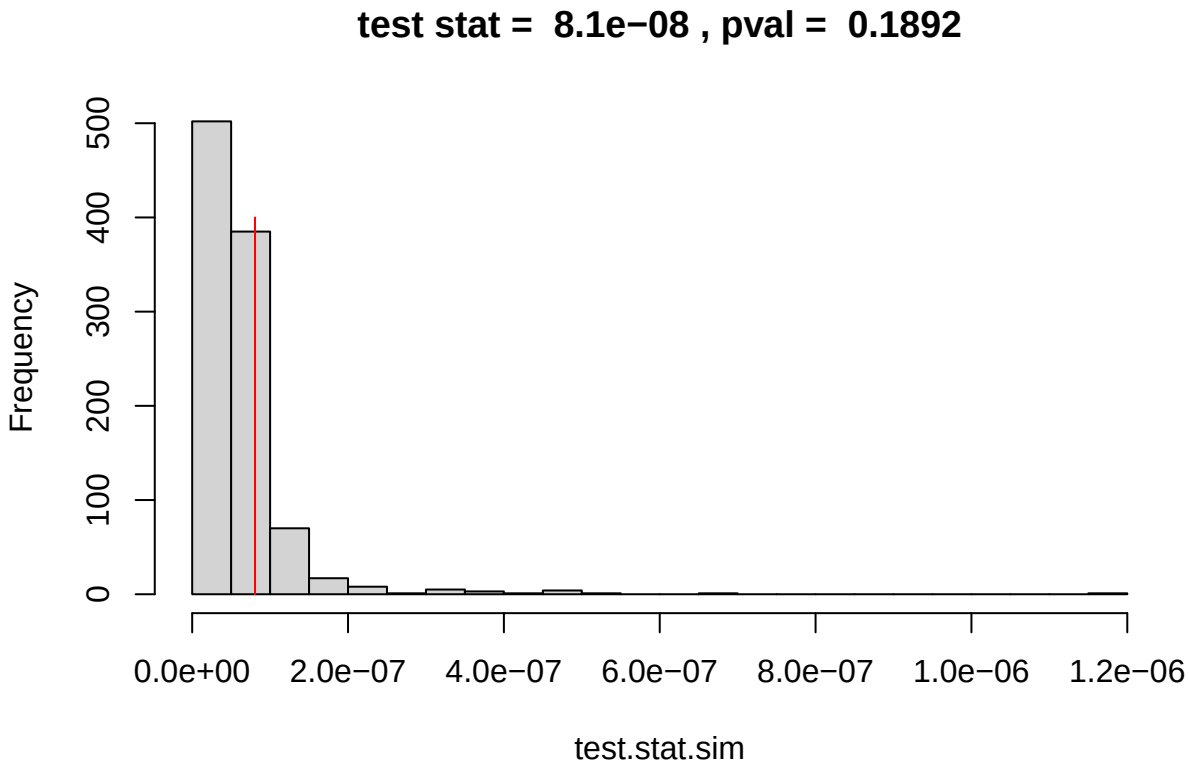
The steps are:

- For each simulation, select 30 locations from the set of all locations to be "cases". (Done by sampling 30 indices from the set of 1:(total number of location), and labeling these "cases" and the others "controls".
- Estimate the case and control intensities, take the ratios and save the values for each grid point.
- After the simulations, we will have (number of simulations) values of the relative risk *at each grid point* under the null distribution.
- We also have the relative risk value from the data *at the same grid points.*
- We use the quantile function to find the 95th and 5th quantiles of the null simulation values *at each grid point.*
- If the data-based relative risk value is $>$ the 95th percentile, write a "+" *at that grid location.*
- If the data-based relative risk value is $<$ the 5th percentile, write a "-" *at that grid location.*

We'll end up with a map showing areas significantly higher (+'s) and significantly lower (-'s) than the null relative risk. These are hot and cold spots, repectively.

```
##################################################
# Now for Kelsall and Diggle tolerance surfaces (identify the grid points that
# are higher than the 97.5th percentile of Monte Carlo surfaces with "+" and
# those that are lower than the 2.5th percentile with "-").
##################################################

# Define the bandwidth
bandw <- 700

# Define the number of simulations
sim <- 999

# 'ind' is an index that goes from 1 to the number of points
ind <- 1:length(dental$x)

# 'rrandmat' is a matrix of zeros with 'sim' rows and a column for each grid point
rrandmat <- matrix(0,sim,grid.number*grid.number)

# 'test.stat.sim' is a vector of zeros to hold the statistic values from each simulation
test.stat.sim <- rep(0,sim)

# Here is the simulation loop
for (i in 1:sim) {

# this is the random label assignment...take a sample of size
# (how many cases are there?) = length of the vector of x-coordinates of the cases
# from 'ind' (the index)
  randind <- sample(ind, length(dental$x[dental$aff==1]))

# define the random labelling cases
  dentrandx <- dental$x[randind]
  dentrandy <- dental$y[randind]
  dentrand.cas.p <- as.points(dentrandx,dentrandy)
# define the random labelling controls
  dentrandx.0 <- dental$x[-randind]
  dentrandy.0 <- dental$y[-randind]
  dentrand.con.p <- as.points(dentrandx.0,dentrandy.0)

# Get the random labelling case density estimate
```

```
  frand <- bkde2D(dentrand.cas.p, casbandw, gridsize=c(grid.number,grid.number),
                  range.x=list( c( min(dentcas.p[,1],dentcon.p[,1])-1.5*casbandw,
                                   max(dentcas.p[,1],dentcon.p[,1])+1.5*casbandw),
                                c( min(dentcas.p[,2],dentcon.p[,2])-1.5*casbandw,
                                   max(dentcas.p[,2],dentcon.p[,2])+1.5*casbandw) )     )

# Get the random labelling control density estimate
  grand <- bkde2D(dentrand.con.p, conbandw, gridsize=c(grid.number,grid.number),
                  range.x=list( c( min(dentcas.p[,1],dentcon.p[,1])-1.5*casbandw,
                                   max(dentcas.p[,1],dentcon.p[,1])+1.5*casbandw),
                                c( min(dentcas.p[,2],dentcon.p[,2])-1.5*casbandw,
                                   max(dentcas.p[,2],dentcon.p[,2])+1.5*casbandw) )     )

  # 'inside' a boolean defined above (T/F)...this sets all values outside the polygon to NA
  # which avoids some divide by Inf and NaN errors.

  frand$fhat[!inside] <- NA
  grand$fhat[!inside] <- NA

# This sets the RR value to the frand value (to preset all of the NAs in the right place)
  rrand <- frand

# Reset the fhat value to the log relative risk value
  rrand$fhat <- log(frand$fhat/grand$fhat)

# calcuate test statistic (omit Inf's and NA's)
  test.stat.sim[i] <- sum( (rrand$fhat[is.finite(rrand$fhat) &                    !is.na(rrand$fhat)]/

# assign row of z values to matrix of simulation values

  for (j in 1:grid.number) {
      rrandmat[i,((j-1)*grid.number+1):(j*grid.number)] <- rrand$fhat[j,]
   }

} # end of sim loop

# Now make a histogram of the test statistics from the random labelling data
# and show the test statistic from the observed data.
#pdf("dental.KD700.hist.pdf",width=11,height=8)
par(mfrow=c(1,1),pty="m")
hist(test.stat.sim,nclass=25,
         main=paste("test stat = ",round(test.stat.obs,9),", pval = ",
         round(length(test.stat.sim[test.stat.sim>test.stat.obs])/
                 length(test.stat.sim),4)))
segments(test.stat.obs,0,test.stat.obs,400,col="red")
```




```
#dev.off()

####################################
# Now to get the tolerance intervals for each grid point
#
# Find indices of non-NA elements  'nona' means it is not an NA
index.nona <- 1:(grid.number^2)

rz.vec <- r.ks[1,]
for (i in 2:grid.number) {
  rz.vec <- c(rz.vec,r.ks[i,])
}

index.nona <- index.nona[!is.na(rz.vec)]

####################################
# Calc quantiles

# First set a matrix that will have the values in it.
rquantmat <- matrix(0,5,length(index.nona))

rrandmat[is.na(rrandmat)] <- -9999

for (j in 1:length(index.nona)) {
  rquantmat[,j] <- quantile(rrandmat[,index.nona[j]],
      c(0.0,0.025,0.5,0.975,1.0))
}

quant.05 <- rep(NA,grid.number*grid.number)
quant.50 <- rep(NA,grid.number*grid.number)
```

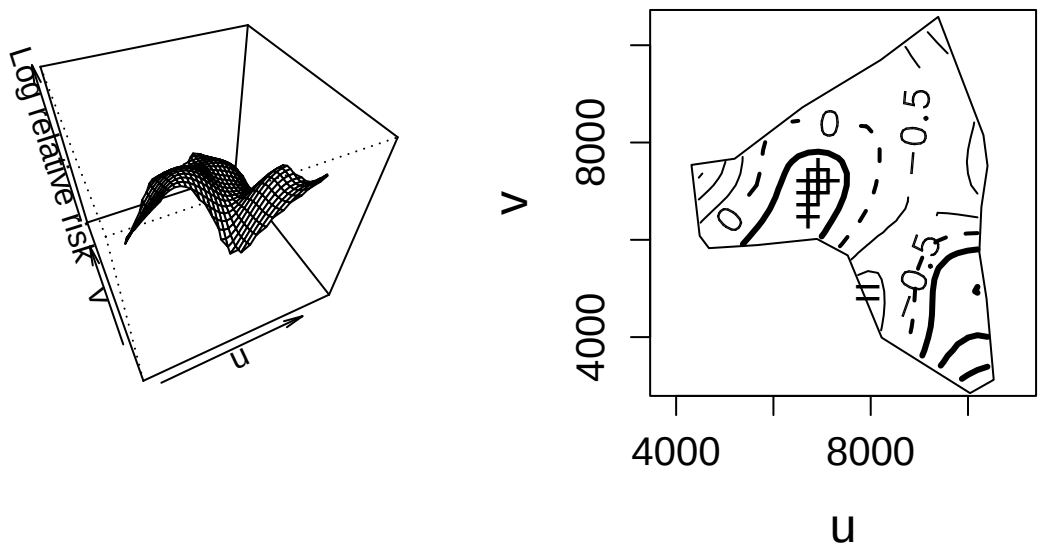
```
quant.95 <- rep(NA,grid.number*grid.number)

quant.05[index.nona] <- rquantmat[2,]
quant.50[index.nona] <- rquantmat[3,]
quant.95[index.nona] <- rquantmat[4,]

quant.05.mat <- matrix(quant.05,grid.number,grid.number)
quant.50.mat <- matrix(quant.50,grid.number,grid.number)
quant.95.mat <- matrix(quant.95,grid.number,grid.number)

# set -9999's to NA's (so they don't map)

quant.05.mat[quant.05.mat==-9999] <- NA

#contour(r$x,r$y,quant.05.mat)
#contour(r$x,r$y,quant.50.mat)
#contour(r$x,r$y,quant.95.mat)

#persp(r$x,r$y,quant.05.mat,theta=30,phi=45)
#persp(r$x,r$y,quant.50.mat,theta=30,phi=45)
#persp(r$x,r$y,quant.95.mat,theta=30,phi=45)
```

Now plot (map) the results!

```
#pdf("dentalKD.tol700.KS.pdf",width=11,height=8)
par(mfrow=c(1,2),pty="s")

    persp(casdens$x1,casdens$x2,r.ks,theta=-25,phi=45,
          xlab="u",ylab="v",zlab="Log relative risk",zlim=c(-3,3))
    title(paste("Log relative risk surface"),cex.main=1.5)

    plot(dentcon.p,pch=" ",xlim=c(min(dental$x)-extra,max(dental$x)+extra),
        ylim=range(dental$y),xlab="u",ylab="v",cex.axis=1.25,cex.lab=1.5)
    contour(casdens$x1,casdens$x2,r.ks,levels=c(-2,-1.5,-1,-0.5,0,0.5,1,1.5,2),
        lwd=c(1,1,1,1,2,3,3,3,3),
        lty=c(1,1,1,1,2,1,1,1,1),
        vfont=c("sans serif","bold"),
        labcex=1.25,add=T )
    polygon(mypoly)
#    points(dentcas.p,pch=16)
#    points(dentcon.p,pch=1)
    title(paste("Gaussian kernel, Bandwidth = ",bandw),cex.main=1.5)


# Create long vectors of (x,y) coords for r

    rx <- rep(casdens$x1,length(casdens$x2))
    for (i in 1:length(casdens$x2)) {
      if (i == 1) {
         ry <- rep(casdens$x2[i],length(casdens$x1))
         }
```

```
     if (i != 1) {
         ry <- c(ry,rep(casdens$x2[i],length(casdens$x1)) )
         }
    }

points(rx[r.ks>quant.95.mat],ry[r.ks>quant.95.mat],pch="+",cex=1.5)
points(rx[r.ks<quant.05.mat],ry[r.ks<quant.05.mat],pch="-",cex=1.5)
```

Log relative risk surfaceœaussian kernel, Bandwidth =




```
#dev,off()
```

Since this example doesn't have strong clustering, the only symbols added are 2 "-" symbols along the lower left edge.