HW10

ASG

2023-04-24

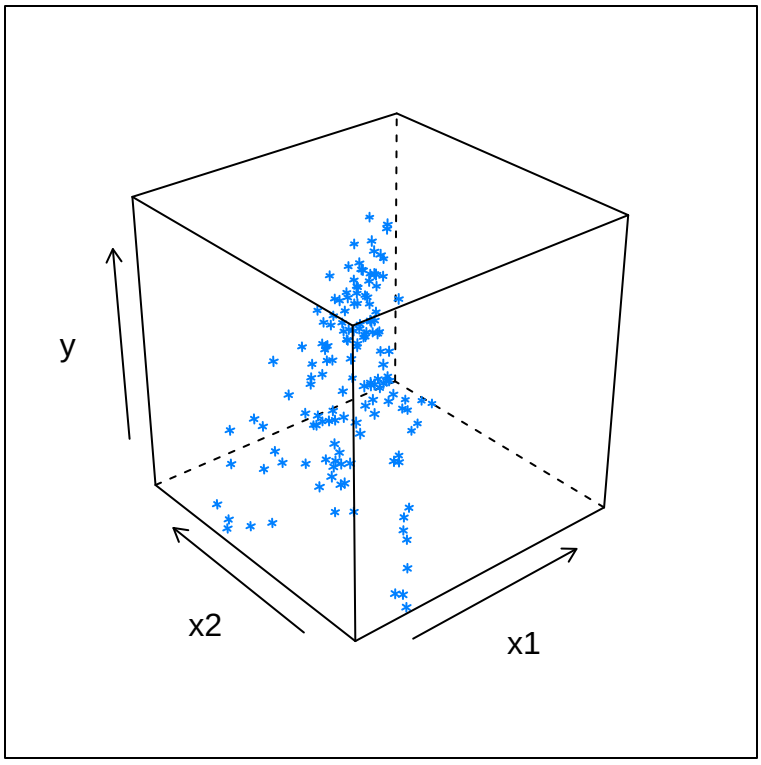
5

a

```
library(HRW) ; data(scallop)
x1 <- scallop$longitude ; x2 <- scallop$latitude
y <- asinh(scallop$totalCatch)
```

b

```
library(lattice) ; cloud(y ~ x1 + x2)
```



c

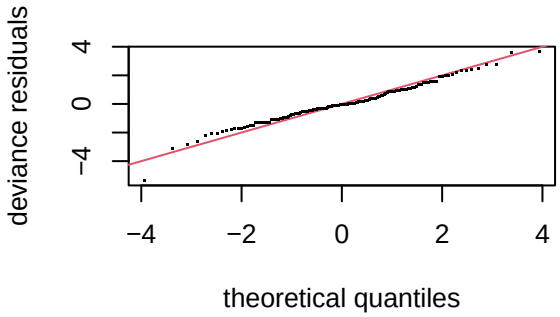
```
library(mgcv) ; fit <- gam(y ~ s(x1,x2))
```

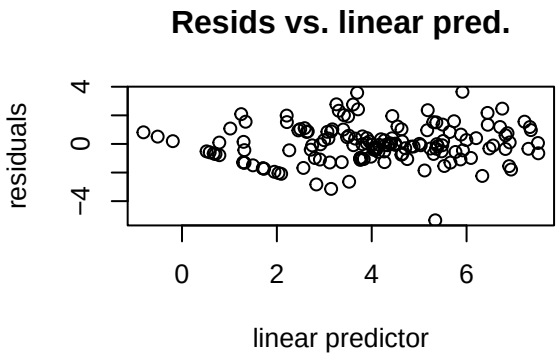
```
## Loading required package: nlme
```

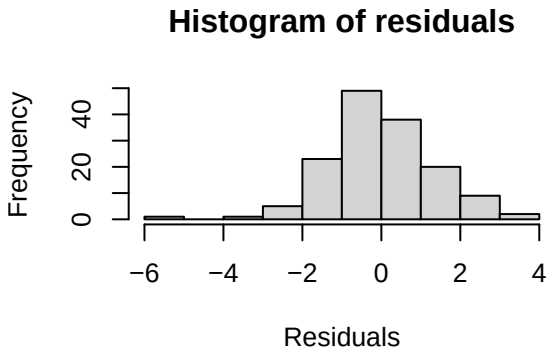
```
## This is mgcv 1.8-42. For overview type 'help("mgcv-package")'.
```

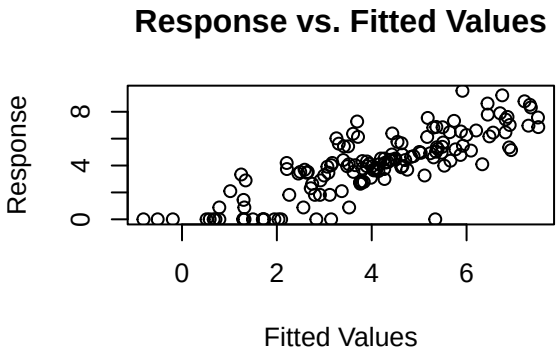
```
summary(fit) ; gam.check(fit)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x1, x2)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.0275     0.1195   33.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##            edf Ref.df     F p-value
## s(x1,x2) 26.18  28.51 8.862  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.624   Deviance explained = 69.1%
## GCV = 2.5875  Scale est. = 2.1123    n = 148
```





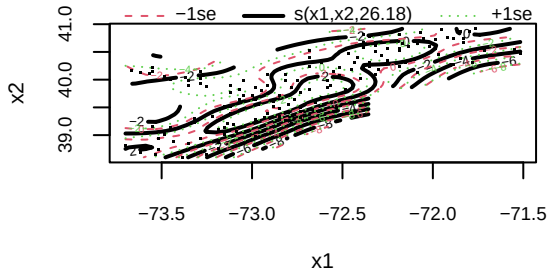


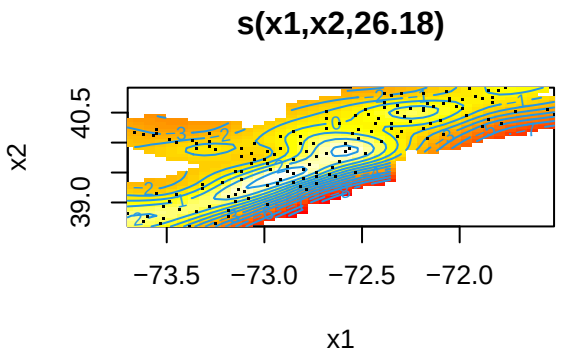


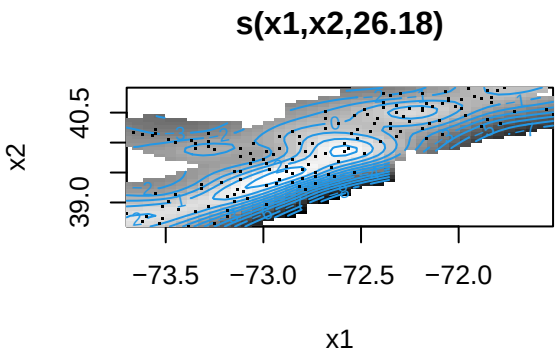

```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradient at convergence was 3.943447e-06 .
## The Hessian was positive definite.
## Model rank =  30 / 30
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##            k'  edf k-index p-value
## s(x1,x2) 29.0 26.2    1.05    0.71
```

d

```
par(mfrow = c(2,2)) ; plot(fit) ; plot(fit,scheme = 1)
plot(fit,scheme = 2) ; plot(fit,scheme = 3)
```







e

```
scallopBdry<-read.table(file="scallopBdry.txt")
print(scallopBdry)
```

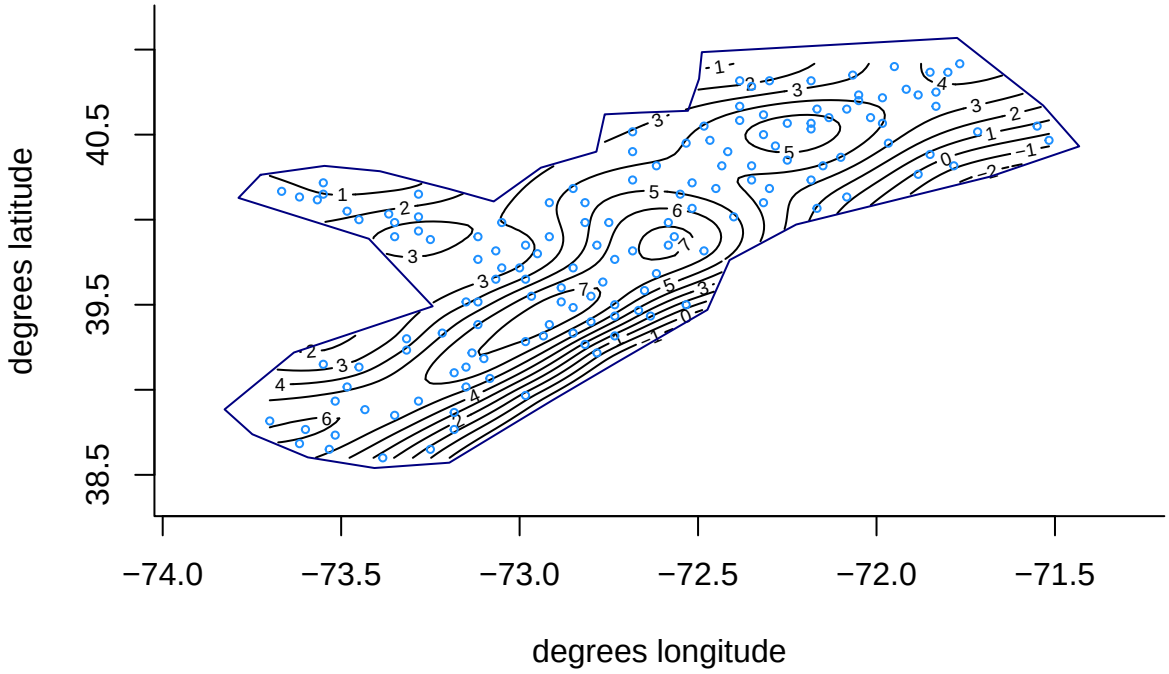
```
##           V1       V2
## 1  -73.72553 40.26403
## 2  -73.54671 40.31627
## 3  -73.39122 40.28493
## 4  -73.19686 40.18044
## 5  -73.07247 40.10731
## 6  -72.94030 40.30582
## 7  -72.78482 40.39986
## 8  -72.76149 40.61927
## 9  -72.66042 40.62972
## 10 -72.52826 40.64017
## 11 -72.49716 40.82823
## 12 -72.48939 40.98496
## 13 -71.77413 41.06854
## 14 -71.53313 40.67151
## 15 -71.43206 40.43120
## 16 -71.68084 40.25358
## 17 -72.22505 39.97148
## 18 -72.41164 39.76251
## 19 -72.47384 39.46996
## 20 -72.72262 39.16696
```

```
## 21 -72.90921 38.93710
## 22 -73.19686 38.57142
## 23 -73.40677 38.54007
## 24 -73.59336 38.60276
## 25 -73.74885 38.73859
## 26 -73.82659 38.88486
## 27 -73.63223 39.21921
## 28 -73.24351 39.49086
## 29 -73.42232 39.88789
## 30 -73.78772 40.12820
## 31 -73.72553 40.26403
```

```
write.table(scallopBdry,"scallopBdry.txt",row.names = FALSE,col.names = FALSE)
scallopBdry<-read.table(file="scallopBdry.txt")
```

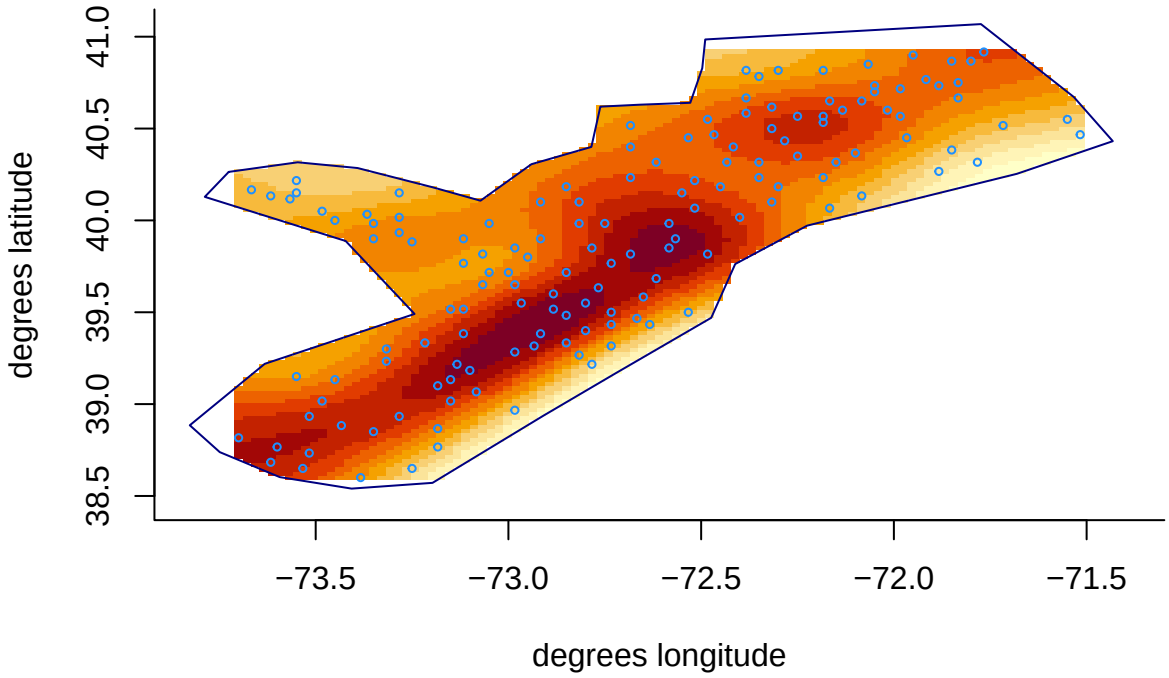
f

```
ngrid <- 101
x1grid <- seq(min(x1),max(x1),length = ngrid)
x2grid <- seq(min(x2),max(x2),length = ngrid)
x1x2mesh <- expand.grid(x1grid,x2grid)
names(x1x2mesh) <- c("x1","x2")
fitmesh <- matrix(predict(fit,newdata = x1x2mesh),ngrid,ngrid)
outInds <- (1:ngrid^2)[pointsInPoly(x1x2mesh,scallopBdry)== FALSE]
fitmesh[outInds] <- NA
xlimVal <- c(1.1*min(x1) - 0.1*max(x1),1.1*max(x1) - 0.1*min(x1))
ylimVal <- c(1.1*min(x2) - 0.1*max(x2),1.1*max(x2) - 0.1*min(x2))
contour(x1grid,x2grid,fitmesh,xlab = "degrees longitude",ylab = "degrees latitude",xlim = xlimVal,ylim =
lines(scallopBdry,col = "navy")
points(x1,x2,col = "dodgerblue",cex = 0.5)
```



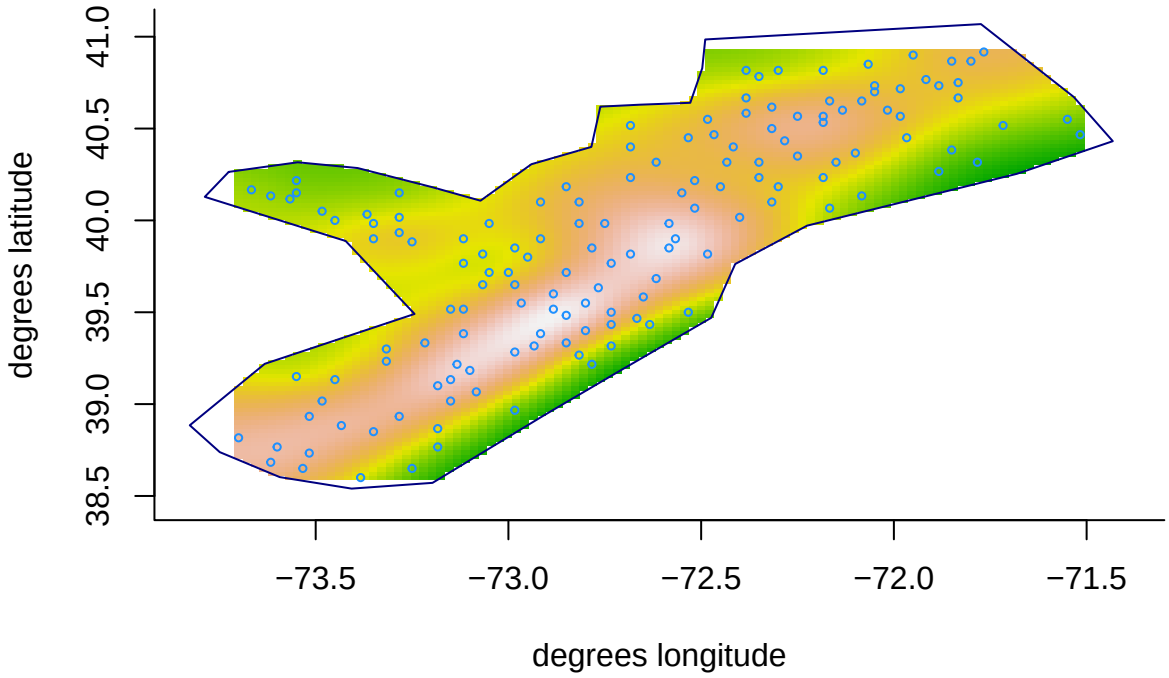
g

```
image(x1grid,x2grid,fitmesh,xlab = "degrees longitude",ylab = "degrees latitude",xlim = xlimVal,ylim = y
lines(scallopBdry,col = "navy")
points(x1,x2,col="dodgerblue",cex=0.5)
```



h

```
image(x1grid,x2grid,fitmesh,col = terrain.colors(1000),xlab = "degrees longitude",ylab = "degrees latitu
lines(scallopBdry,col = "navy")
points(x1,x2,col="dodgerblue",cex=0.5)
```

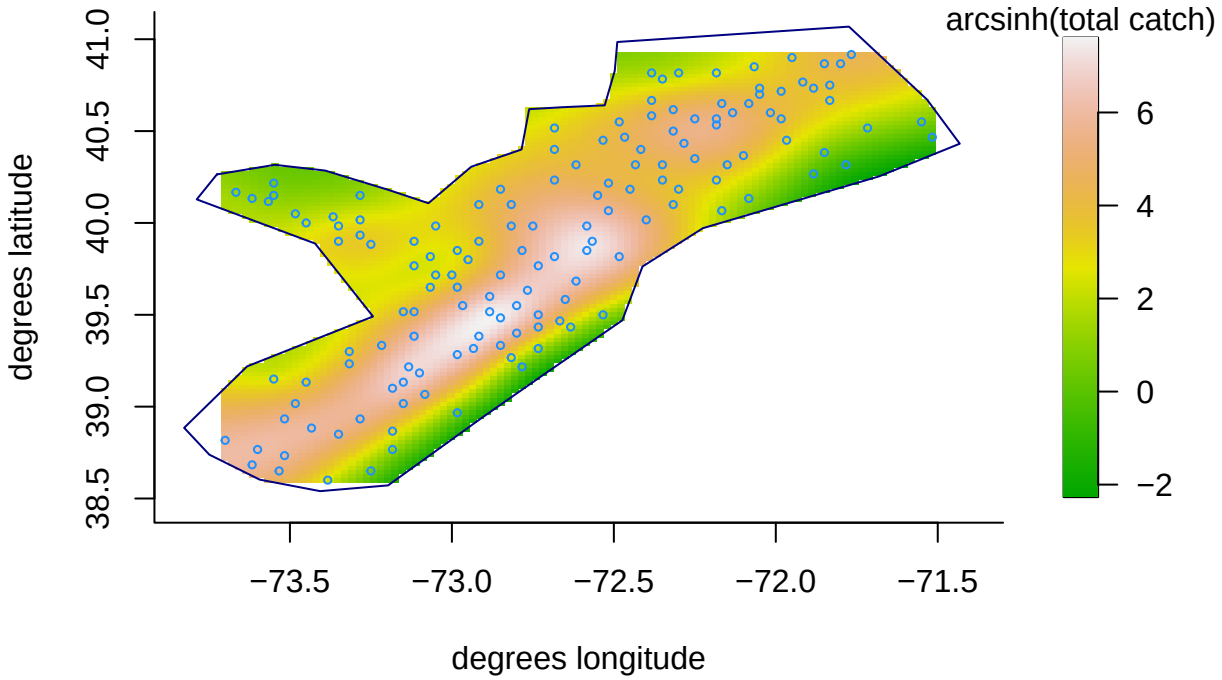


1

```
library(fields)
```

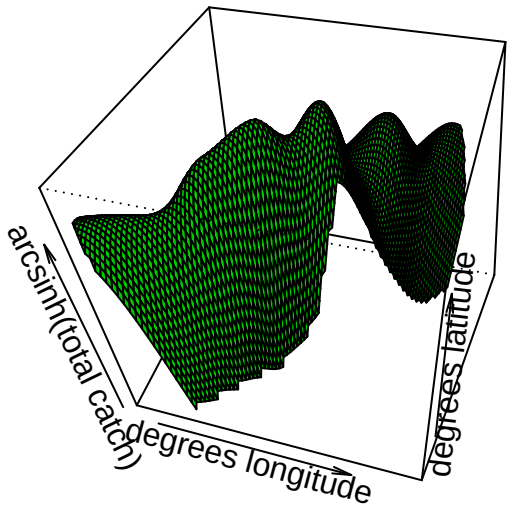
```
## Loading required package: spam

## Spam version 2.9-1 (2022-08-07) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

##
## Attaching package: 'spam'

## The following objects are masked from 'package:base':
##
##     backsolve, forwardsolve

## Loading required package: viridis

## Loading required package: viridisLite

##
## Try help(fields) to get started.
```

```
image.plot(x1grid,x2grid,fitmesh,col = terrain.colors(1000),xlab = "degrees longitude",ylab = "degrees l
lines(scallopBdry,col = "navy")
points(x1,x2,col="dodgerblue",cex=0.5)
```



j

```
persp(x1grid,x2grid,fitmesh,col = "green3",theta = 15,
phi = 45,xlab = "degrees longitude",ylab = "degrees latitude",zlab = "arcsinh(total catch)")
```



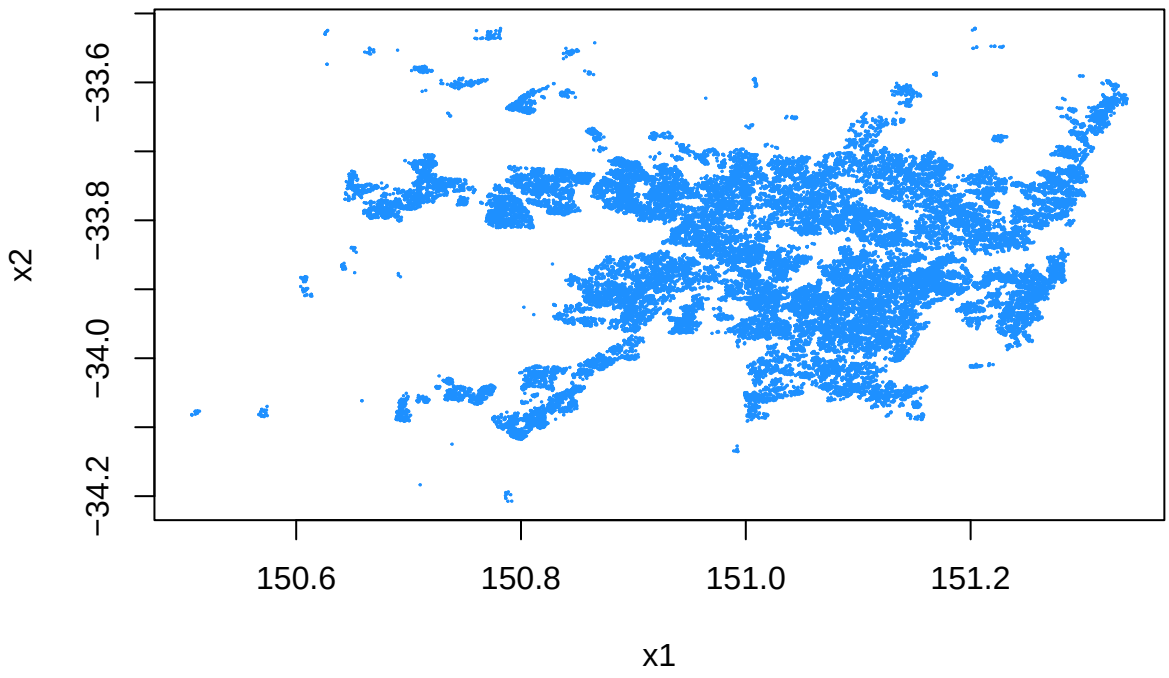
5.2

a

```
library(HRW) ; data(SydneyRealEstate)
x1 <- SydneyRealEstate$longitude
x2 <- SydneyRealEstate$latitude
y <- SydneyRealEstate$logSalePrice
```

b

```
plot(x1,x2,col = "dodgerblue",cex = 0.1)
```



c

```
plot(x1,x2,col = "dodgerblue",cex = 0.1)
data(SydneyRealEstateBdry)
print(SydneyRealEstateBdry)
```

```
##     longitude  latitude
## 1    151.3427 -33.61223
## 2    151.3357 -33.64283
## 3    151.3226 -33.67875
## 4    151.3105 -33.68407
## 5    151.3115 -33.70535
## 6    151.3065 -33.71999
## 7    151.3045 -33.75325
## 8    151.3025 -33.77453
## 9    151.2934 -33.79050
## 10   151.2924 -33.80779
## 11   151.2773 -33.80912
## 12   151.2672 -33.81710
## 13   151.2551 -33.81976
## 14   151.2612 -33.83440
## 15   151.2421 -33.84637
## 16   151.2270 -33.85036
## 17   151.2078 -33.84903
## 18   151.1947 -33.84504
## 19   151.1887 -33.84105
```

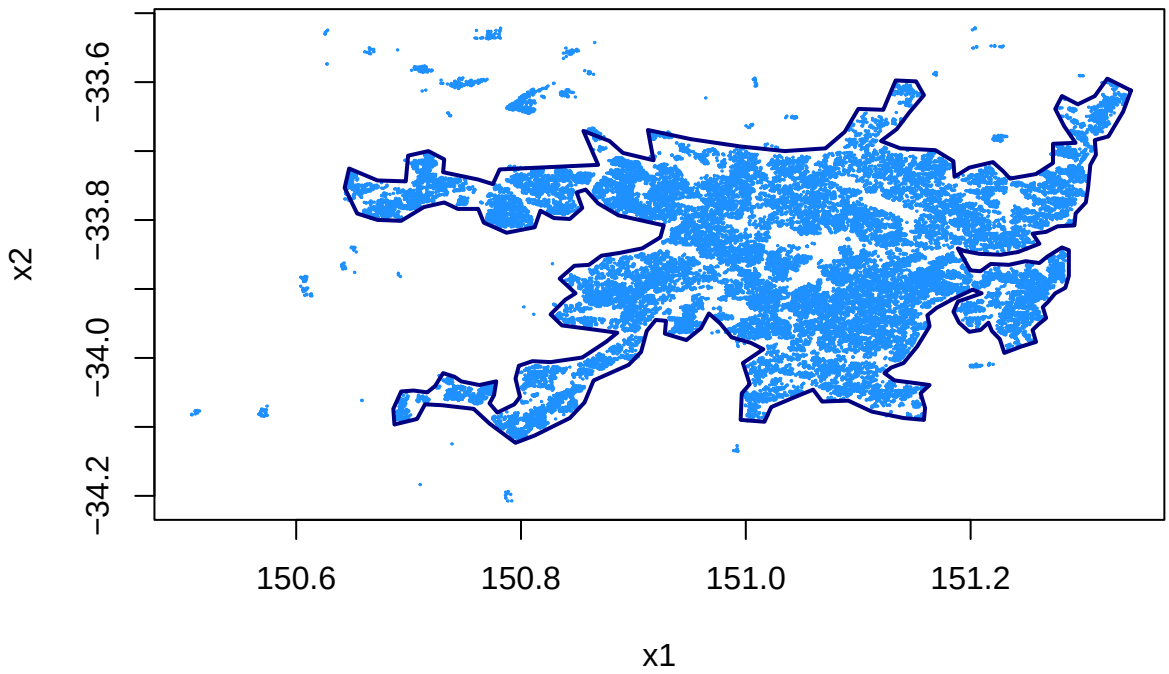
```
## 20   151.1937 -33.85568
## 21   151.1998 -33.87298
## 22   151.2088 -33.87431
## 23   151.2179 -33.86367
## 24   151.2340 -33.86500
## 25   151.2491 -33.85967
## 26   151.2612 -33.86234
## 27   151.2682 -33.85302
## 28   151.2813 -33.83972
## 29   151.2874 -33.84371
## 30   151.2874 -33.86234
## 31   151.2874 -33.88096
## 32   151.2843 -33.89825
## 33   151.2753 -33.90624
## 34   151.2692 -33.91821
## 35   151.2642 -33.92619
## 36   151.2672 -33.94216
## 37   151.2602 -33.95147
## 38   151.2551 -33.95945
## 39   151.2582 -33.97675
## 40   151.2451 -33.98340
## 41   151.2300 -33.99271
## 42   151.2260 -33.97275
## 43   151.2189 -33.96078
## 44   151.2159 -33.94881
## 45   151.2088 -33.95945
## 46   151.1988 -33.96211
## 47   151.1897 -33.94881
## 48   151.1847 -33.93284
## 49   151.1887 -33.91821
## 50   151.2098 -33.90624
## 51   151.2018 -33.90092
## 52   151.1847 -33.91422
## 53   151.1696 -33.92752
## 54   151.1615 -33.93817
## 55   151.1635 -33.95413
## 56   151.1525 -33.98340
## 57   151.1404 -34.00734
## 58   151.1293 -34.01399
## 59   151.1233 -34.02198
## 60   151.1323 -34.03262
## 61   151.1454 -34.03528
## 62   151.1635 -34.03927
## 63   151.1555 -34.05124
## 64   151.1595 -34.07253
## 65   151.1585 -34.08982
## 66   151.1404 -34.08716
## 67   151.1122 -34.07785
## 68   151.0911 -34.06189
## 69   151.0679 -34.06322
## 70   151.0599 -34.04592
## 71   151.0458 -34.05524
## 72   151.0226 -34.07120
## 73   151.0166 -34.09249
```

```
## 74   150.9954 -34.08982
## 75   150.9964 -34.05124
## 76   151.0035 -34.03794
## 77   150.9974 -34.00734
## 78   151.0156 -33.98739
## 79   151.0045 -33.97808
## 80   150.9874 -33.97009
## 81   150.9773 -33.95014
## 82   150.9672 -33.93550
## 83   150.9602 -33.95679
## 84   150.9471 -33.97408
## 85   150.9280 -33.96477
## 86   150.9290 -33.94615
## 87   150.9199 -33.94482
## 88   150.9119 -33.96078
## 89   150.9068 -33.99138
## 90   150.8958 -34.01000
## 91   150.8646 -34.03262
## 92   150.8565 -34.06455
## 93   150.8434 -34.08716
## 94   150.8122 -34.11244
## 95   150.7951 -34.12308
## 96   150.7720 -34.09515
## 97   150.7579 -34.07386
## 98   150.7297 -34.06854
## 99   150.7146 -34.06721
## 100  150.7075 -34.08849
## 101  150.6874 -34.09648
## 102  150.6864 -34.07386
## 103  150.6934 -34.04858
## 104  150.7045 -34.04725
## 105  150.7166 -34.04991
## 106  150.7236 -34.04060
## 107  150.7307 -34.02198
## 108  150.7408 -34.02730
## 109  150.7468 -34.03395
## 110  150.7629 -34.03927
## 111  150.7780 -34.03395
## 112  150.7760 -34.05391
## 113  150.7720 -34.06588
## 114  150.7790 -34.07918
## 115  150.7941 -34.06721
## 116  150.7991 -34.05657
## 117  150.7951 -34.02996
## 118  150.7981 -34.01133
## 119  150.8102 -34.00468
## 120  150.8263 -34.00601
## 121  150.8545 -33.99936
## 122  150.8756 -33.97675
## 123  150.8857 -33.96344
## 124  150.8616 -33.95812
## 125  150.8364 -33.95280
## 126  150.8263 -33.93683
## 127  150.8394 -33.91555
```

```
## 128  150.8485 -33.90624
## 129  150.8344 -33.88495
## 130  150.8475 -33.86633
## 131  150.8605 -33.86500
## 132  150.8716 -33.85169
## 133  150.8877 -33.84770
## 134  150.9079 -33.84105
## 135  150.9240 -33.82509
## 136  150.9270 -33.80779
## 137  150.8867 -33.79316
## 138  150.8686 -33.77586
## 139  150.8575 -33.75591
## 140  150.8495 -33.75990
## 141  150.8545 -33.78251
## 142  150.8434 -33.79848
## 143  150.8293 -33.79715
## 144  150.8173 -33.78651
## 145  150.8122 -33.81045
## 146  150.7871 -33.81843
## 147  150.7669 -33.80380
## 148  150.7619 -33.78384
## 149  150.7438 -33.78384
## 150  150.7317 -33.77453
## 151  150.7136 -33.78118
## 152  150.6934 -33.80114
## 153  150.6723 -33.79981
## 154  150.6542 -33.79050
## 155  150.6431 -33.75325
## 156  150.6471 -33.72531
## 157  150.6723 -33.74260
## 158  150.6975 -33.74393
## 159  150.6995 -33.70668
## 160  150.7176 -33.70003
## 161  150.7317 -33.71201
## 162  150.7307 -33.73063
## 163  150.7619 -33.74127
## 164  150.7750 -33.74793
## 165  150.7810 -33.72664
## 166  150.8686 -33.71999
## 167  150.8555 -33.67077
## 168  150.8787 -33.68540
## 169  150.8907 -33.70269
## 170  150.9179 -33.71334
## 171  150.9129 -33.66944
## 172  150.9511 -33.68274
## 173  150.9954 -33.69338
## 174  151.0347 -33.70003
## 175  151.0709 -33.69604
## 176  151.0880 -33.67210
## 177  151.1001 -33.63884
## 178  151.1223 -33.64017
## 179  151.1333 -33.59760
## 180  151.1515 -33.59893
## 181  151.1585 -33.61888
```

```
## 182  151.1454 -33.64416
## 183  151.1343 -33.66810
## 184  151.1203 -33.68540
## 185  151.1374 -33.69604
## 186  151.1686 -33.69870
## 187  151.1847 -33.71467
## 188  151.1857 -33.73728
## 189  151.1988 -33.72398
## 190  151.2199 -33.71600
## 191  151.2290 -33.72930
## 192  151.2350 -33.73994
## 193  151.2582 -33.73329
## 194  151.2733 -33.71733
## 195  151.2733 -33.68939
## 196  151.2934 -33.68806
## 197  151.2833 -33.66411
## 198  151.2753 -33.63884
## 199  151.2813 -33.62021
## 200  151.2954 -33.63219
## 201  151.3105 -33.62021
## 202  151.3216 -33.59494
## 203  151.3427 -33.61223
```

```
lines(SydneyRealEstateBdry,col = "navy",lwd = 2)
```



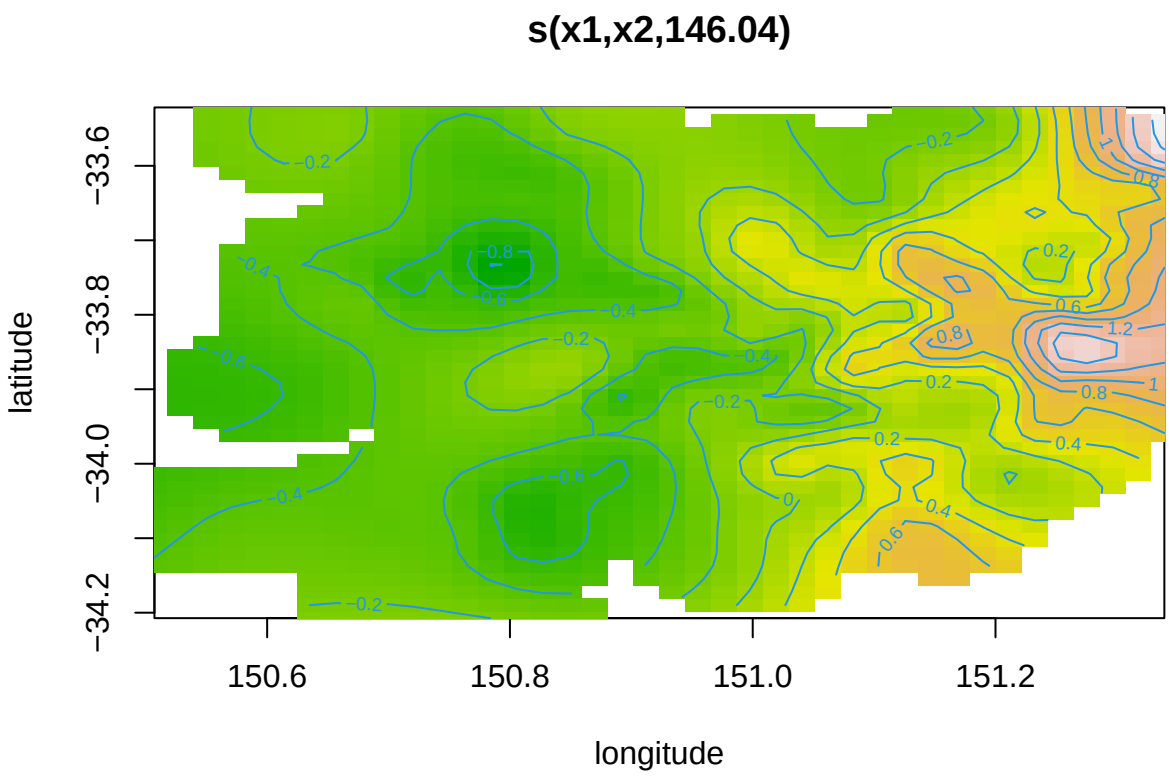
d

```
library(mgcv) ; fit <- gam(y ~ s(x1,x2,k = 150))
summary(fit)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x1, x2, k = 150)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13.079340   0.001764    7415   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df     F p-value
## s(x1,x2) 146  148.9 490.9  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.66   Deviance explained = 66.1%
## GCV = 0.11769  Scale est. = 0.11723   n = 37676
```

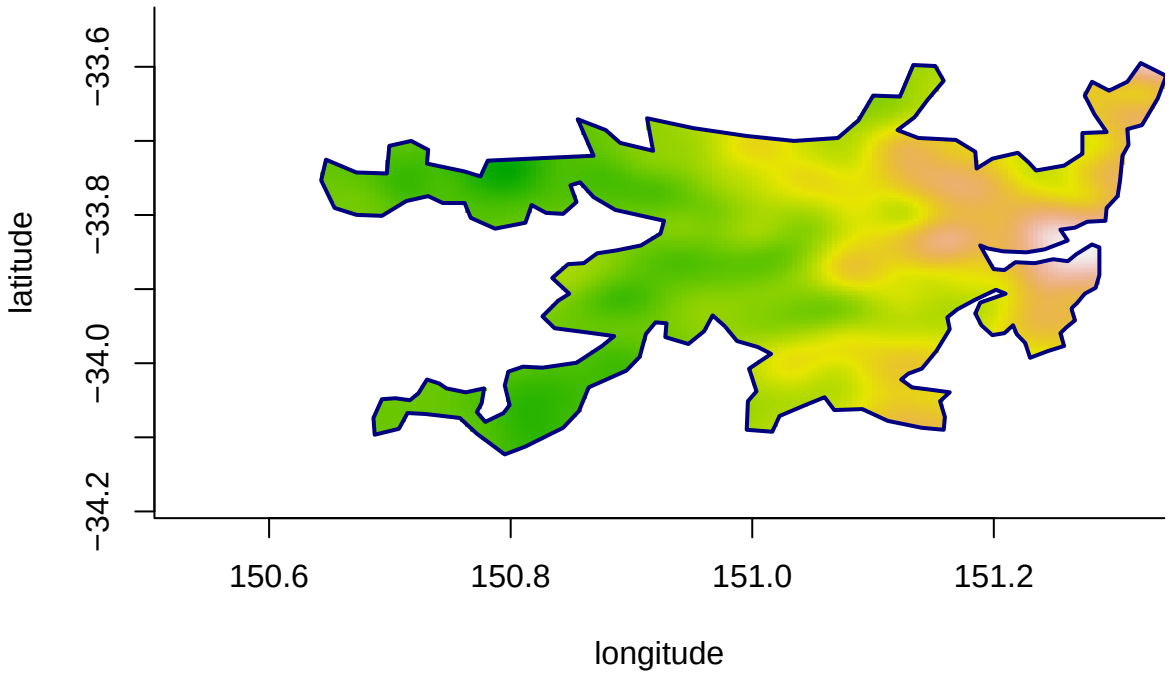
e

```
plot(fit,scheme = 2,hcolors = terrain.colors(1000),xlab = "longitude",ylab = "latitude")
```



f

```
ngrid <- 201
x1grid <- seq(min(x1),max(x1),length = ngrid)
x2grid <- seq(min(x2),max(x2),length = ngrid)
x1x2mesh <- expand.grid(x1grid,x2grid)
names(x1x2mesh) <- c("x1","x2")
fitmesh <- predict(fit,newdata = x1x2mesh)
outInds <- (1:ngrid^2)[pointsInPoly(x1x2mesh,SydneyRealEstateBdry) == FALSE]
fitmesh[outInds] <- NA
fitmesh <- matrix(fitmesh,ngrid,ngrid)
image(x1grid,x2grid,fitmesh,col = terrain.colors(1000),xlab = "longitude",ylab = "latitude",bty = "l")
lines(SydneyRealEstateBdry,col = "navy",lwd = 2)
```



3

The convex hull of the spatial data does not always lead to a good region for switching on pixels in image plots. There are instances when the data is far from convex and thus it is not a good pixel switch on boundary. Hence plotting boundaries are integral in plotting 2d smoothing function.