Criterion behaviour

B. Charnomordic

2017-09-12

Contents


```
library(geozoning)
```

This vignette illustrates the criterion behaviour on 2 different zonings in 4 steps. First a simulated map is generated, second an exploraty loop is run to rank the zonings for a size 3 quantile vector (4-label map). Then the correction procedure is run independently for the best and the worst zonings found in that exploratory loop. The criterion values are printed and explained.

Step 1: Generate map

A map object is simulated with a Gaussian field and a variogram model. 450 points are randomly allocated on a square field of size 1.Then 1936 points are kriged on a regular grid using inverse distance weighted interpolation. A Delaunay tesselation yields point neighborhood in the sense of Voronoi.

```
seed=10
map=genMap(DataObj=NULL,seed=seed,disp=FALSE,krig=1)
```

```
## [1] "DataObj=NULL, generating DataObj-seed= 10"
## [using ordinary kriging]
```

Step 2: Exploratory loop for a size 3 quantile vector (4-label map)

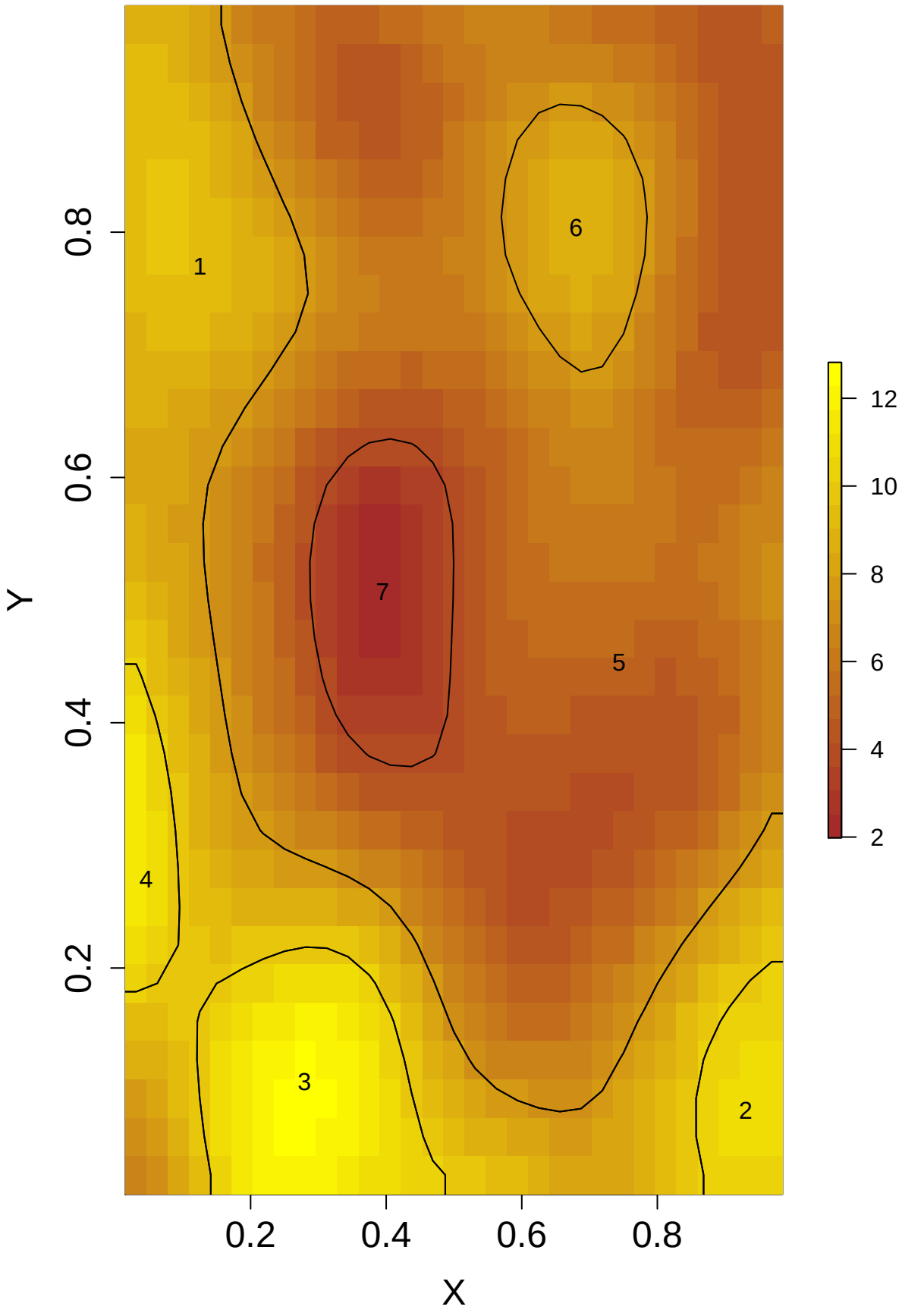
```
ro=loopQ3(map,step=0.1,disp=0,QUIET=T)
```

ro is a matrix sorted by reverse order of crit. It has 8 columns and 56 rows. Columns contain the following values calculated for each quantile vector: criterion, cost, cost per label, number of zones, quantile associated probability values and number of non degenerated quantiles. Each row corresponds to the best zoning obtained for the corresponding quantile at the end of the correction procedure. To save time and memory, details are not saved.

Step 3: Run correctionTree procedure for best zoning and save results

```
bqProb=ro[1,5:7]
criti=correctionTree(bqProb,map,SAVE=TRUE)
res=searchNODcrit1(bqProb,criti)
b=res$ind[[1]][1]
K=criti$zk[[2]][[b]]
```

```
bZ=K$zonePolygone
dispZ(map$step,map$krigGrid,zonePolygone=bZ)
```


Y
0.2
0.4
0.6
0.8
X
0.2
0.4
0.6
0.8
1
2
3
4
5
6
7
12
10
8
6
4
2

```
## NULL
```

```
  # distance matrix has high values, criterion is the smallest one (6.417), distance between zones 5 an
  bmd=criti$mdist[[2]][[b]]
  bcrit=criti$criterion[[2]][[b]]
  bcrit
```

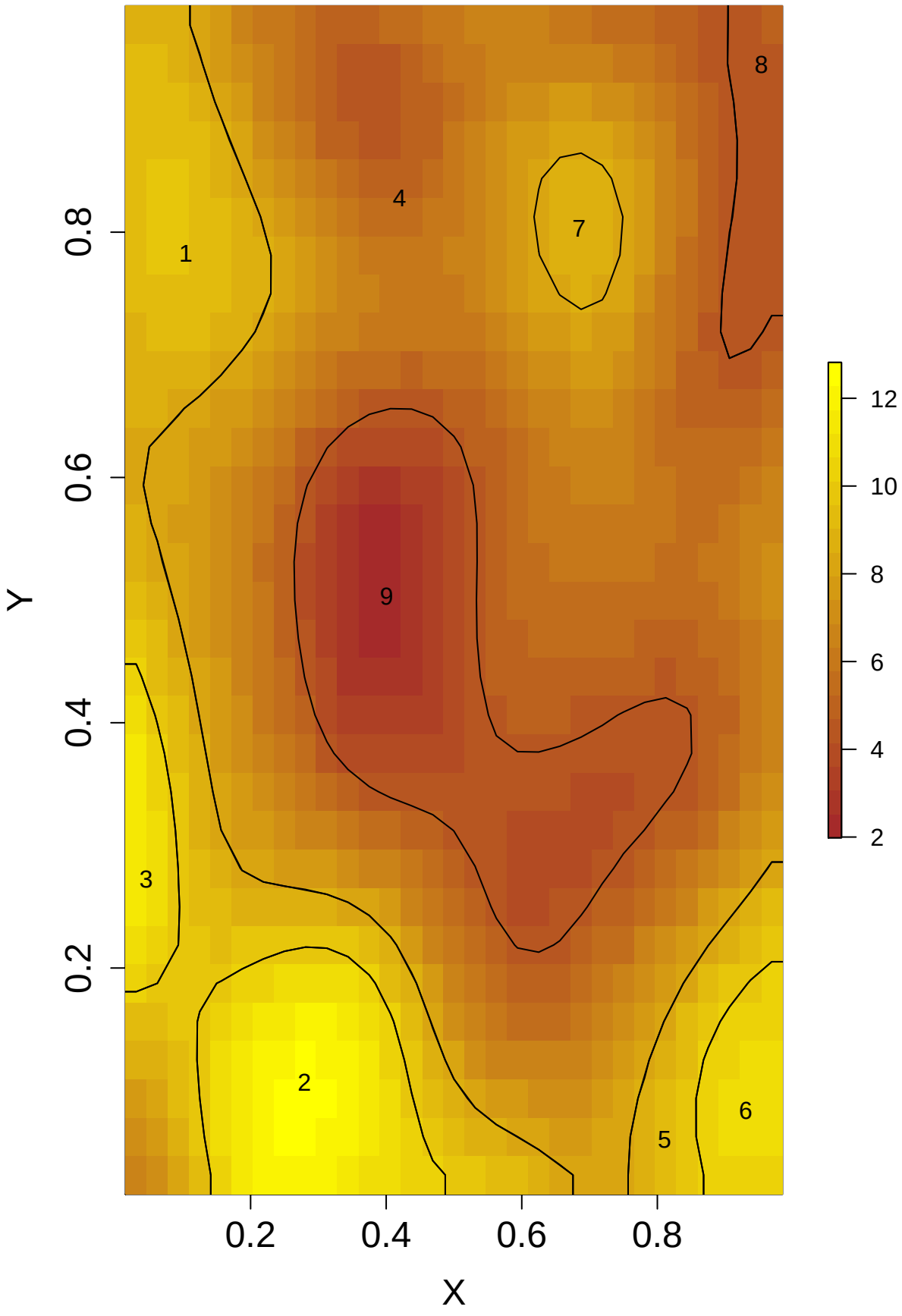
```
## [1] 6.416577
```

```
  bmd
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 1.000000 6.506939 6.868417 6.554862 7.642284 0.000000 0.000000
## [2,] 6.506939 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000
## [3,] 6.868417 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000
## [4,] 6.554862 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000
## [5,] 7.642284 0.000000 0.000000 0.000000 1.000000 6.953671 6.416577
## [6,] 0.000000 0.000000 0.000000 0.000000 6.953671 1.000000 0.000000
## [7,] 0.000000 0.000000 0.000000 0.000000 6.416577 0.000000 1.000000
```

Step 4: Run correctionTree procedure for worst zoning and save results

```
  wqProb=ro[56,5:7]
  criti=correctionTree(wqProb,map,SAVE=TRUE)
  res=searchNODcrit1(wqProb,criti)
  w=res$ind[[1]][1]
  K=criti$zk[[2]][[w]]
  wZ=K$zonePolygone
  dispZ(map$step,map$krigGrid,zonePolygone=wZ)
```


8
4
7
1
9
3
2
6
5
0.2
0.4
0.6
0.8
Y
0.2
0.4
0.6
0.8
X
12
10
8
6
4
2

```
## NULL
```

```
  # distance matrix has some low values, criterion is the smallest one (3.747), distance between zones
  wmd=criti$mdist[[2]][[w]]
  wcrit=criti$criterion[[2]][[w]]
  wcrit
```

```
## [1] 3.861315
```

```
  wmd
```

```
##           [,1]     [,2]    [,3]     [,4]     [,5]     [,6]     [,7]
##  [1,] 1.000000 6.181142 5.89439 6.051933 0.000000 0.000000 0.000000
##  [2,] 6.181142 1.000000 0.00000 0.000000 0.000000 0.000000 0.000000
##  [3,] 5.894390 0.000000 1.00000 0.000000 0.000000 0.000000 0.000000
##  [4,] 6.051933 0.000000 0.00000 1.000000 6.974534 0.000000 5.927093
##  [5,] 0.000000 0.000000 0.00000 6.974534 1.000000 8.139785 0.000000
##  [6,] 0.000000 0.000000 0.00000 0.000000 8.139785 1.000000 0.000000
##  [7,] 0.000000 0.000000 0.00000 5.927093 0.000000 0.000000 1.000000
##  [8,] 0.000000 0.000000 0.00000 3.861315 0.000000 0.000000 0.000000
##  [9,] 0.000000 0.000000 0.00000 4.814187 0.000000 0.000000 0.000000
##           [,8]     [,9]
##  [1,] 0.000000 0.000000
##  [2,] 0.000000 0.000000
##  [3,] 0.000000 0.000000
##  [4,] 3.861315 4.814187
##  [5,] 0.000000 0.000000
##  [6,] 0.000000 0.000000
##  [7,] 0.000000 0.000000
##  [8,] 1.000000 0.000000
##  [9,] 0.000000 1.000000
```

Session informations

```
## R version 3.4.0 (2017-04-21)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Debian GNU/Linux 8 (jessie)
##
## Matrix products: default
## BLAS: /usr/lib/libblas/libblas.so.3.0
## LAPACK: /usr/lib/lapack/liblapack.so.3.0
##
## locale:
##  [1] LC_CTYPE=fr_FR.utf8       LC_NUMERIC=C
##  [3] LC_TIME=fr_FR.utf8        LC_COLLATE=fr_FR.utf8
##  [5] LC_MONETARY=fr_FR.utf8    LC_MESSAGES=fr_FR.utf8
##  [7] LC_PAPER=fr_FR.utf8       LC_NAME=C
##  [9] LC_ADDRESS=C              LC_TELEPHONE=C
## [11] LC_MEASUREMENT=fr_FR.utf8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
```

```
## [1] geozoning_1.0.0 rmarkdown_1.6
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.11             raster_2.5-8
##  [3] knitr_1.17               magrittr_1.5
##  [5] maptools_0.9-2           maps_3.1.1
##  [7] lattice_0.20-35          FNN_1.1
##  [9] stringr_1.2.0            xts_0.9-7
## [11] fields_8.15              tools_3.4.0
## [13] RandomFields_3.1.50      grid_3.4.0
## [15] gstat_1.1-5              spam_1.4-0
## [17] deldir_0.1-14            intervals_0.15.1
## [19] RandomFieldsUtils_0.3.25 htmltools_0.3.6
## [21] rgeos_0.3-23             yaml_2.1.14
## [23] rprojroot_1.2            digest_0.6.12
## [25] evaluate_0.10.1          sp_1.2-4
## [27] stringi_1.1.5            compiler_3.4.0
## [29] backports_1.1.0          spacetime_1.2-0
## [31] foreign_0.8-68           zoo_1.8-0
```