

Postprocessing zoning: smoothing zones

B. Charnomordic

2017-09-26

Contents

Session informations

7

```
library(geozoning)
library(rgeos)
```

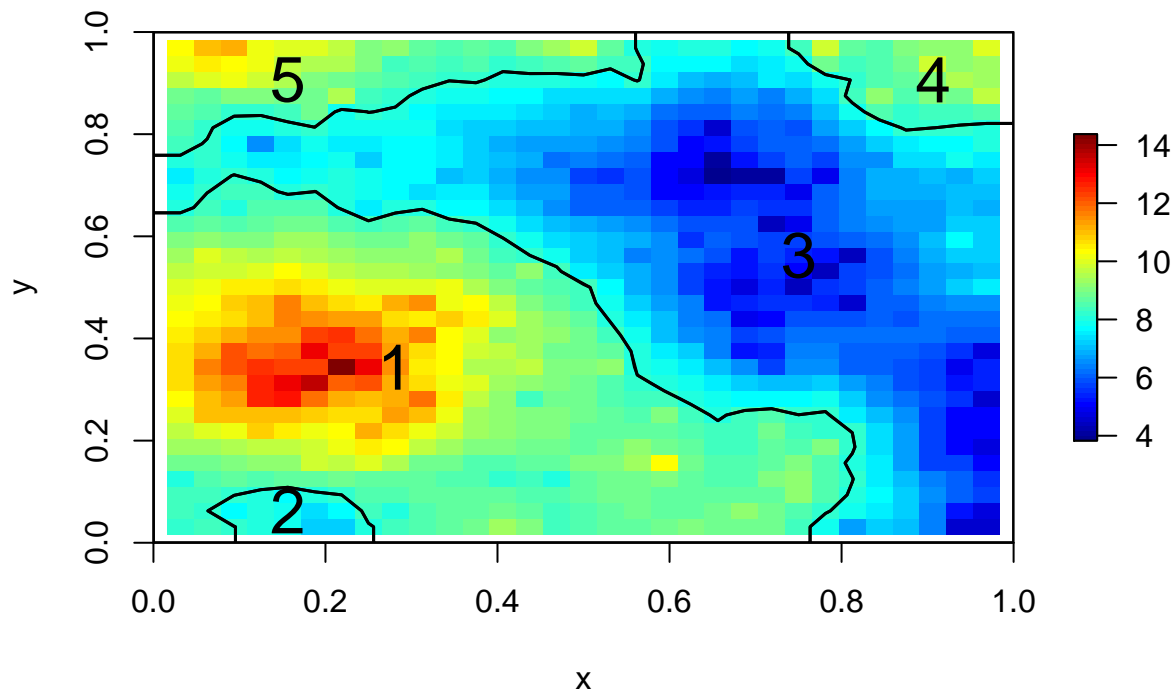
This vignette illustrates the zone and map smoothing process. It first shows why a step of zone boundary correction is required. Then the zone smoothing based on morphological erosion and dilatation is illustrated. Finally the map smothing resulting of all zone smoothing is illustrated.

Step 1: Why is CorrectBoundaryMap required

```
seed=1
map=genMap(DataObj=NULL,seed=seed,disp=FALSE,krig=2)

## [1] "DataObj=NULL, generating DataObj-seed= 1"
## [inverse distance weighted interpolation]

criti = correctionTree(qProb = c(0.5), map = map)
Z = criti$zk[[1]][[1]]$zonePolygone
lab = criti$zk[[1]][[1]]$lab
plotM(map = map, Z = Z, lab = lab, byLab = FALSE)
```



```
class(gIntersection(Z[[1]],Z[[2]])) [1]
```

```
## [1] "SpatialPolygons"
```

```

class(gIntersection(Z[[1]],Z[[5]])) [1]

## [1] "NULL"
class(gIntersection(Z[[2]],Z[[3]])) [1]

## [1] "NULL"
class(gIntersection(Z[[2]],Z[[4]])) [1]

## [1] "NULL"
res = correctBoundaryMap(Zi = Z, map = map)
Z = res$Z
class(gIntersection(Z[[1]],Z[[2]])) [1]

## [1] "SpatialLines"
class(gIntersection(Z[[1]],Z[[5]])) [1]

## [1] "NULL"
class(gIntersection(Z[[2]],Z[[3]])) [1]

## [1] "NULL"
class(gIntersection(Z[[2]],Z[[4]])) [1]

## [1] "NULL"
Step 2: Smoothing a zone
seed=1

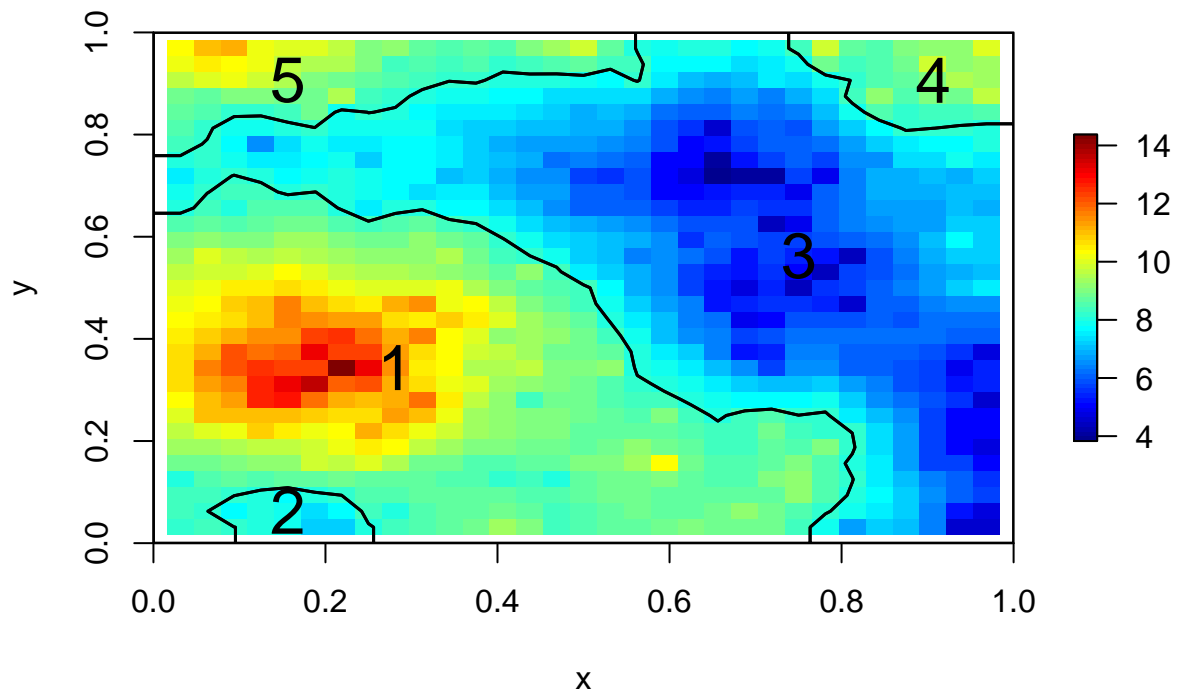
Generate map with simulated data
map=genMap(DataObj=NULL,seed=seed,disp=FALSE,krig=2)

## [1] "DataObj=NULL, generating DataObj-seed= 1"
## [inverse distance weighted interpolation]

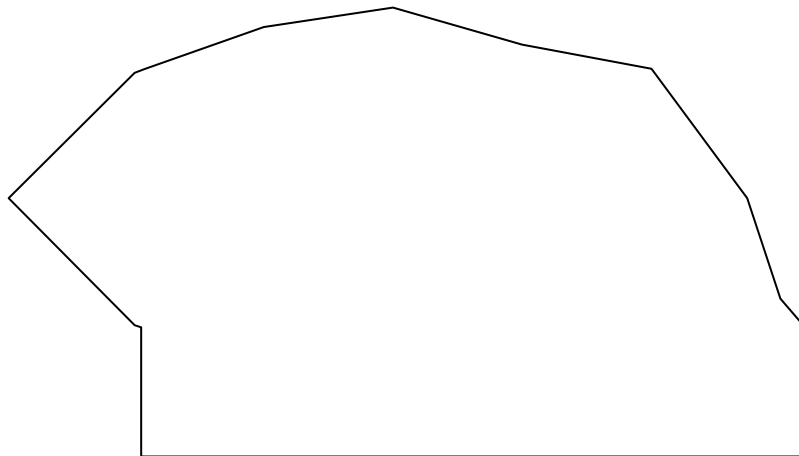
Generate zoning
criti = correctionTree(qProb = c(0.5), map = map)
Z = criti$zk[[1]][[1]]$zonePolygone
lab = criti$zk[[1]][[1]]$lab

Correct zone boundaries
res = correctBoundaryMap(Zi = Z, map = map)
Z = res$Z
# map boundary after correction
boundary = Z[[1]]
for(i in 2:length(Z)){
  boundary = gUnion(boundary, Z[[i]])
}
#plot map
plotM(map = map, Z = Z, lab = lab, byLab = FALSE)

```



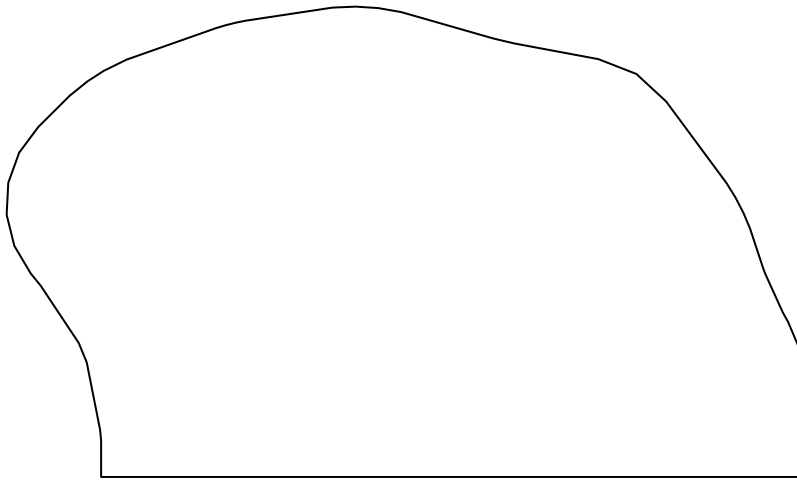
```
# smoothing
zone = Z[[2]]
plot(zone)
```



```
newZone = smoothingZone(z = zone, width = 0.05, boundary = boundary)
```

```
## [1] "widthExt 0.05"
## [1] "widthInt 0.025"
```

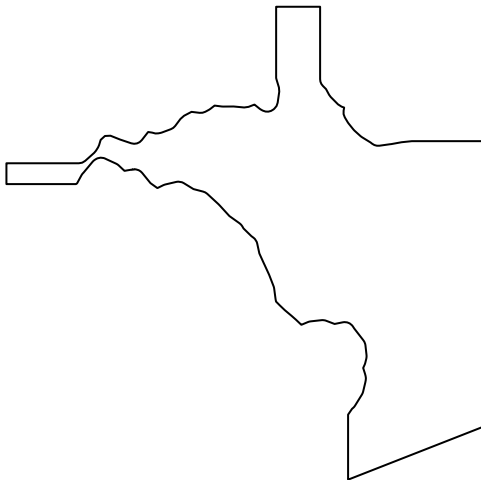
```
plot(newZone)
```



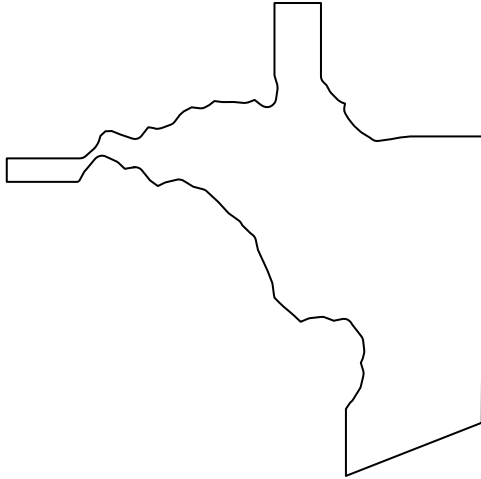
Compute maximum width for zone

smoothing

```
widthMax = cal.max.width.Zone(z = Z[[3]], step = 0.001, widthMax = 0.05, boundary = boundary, erosion =
zone = zone.extended(z = Z[[3]], boundary = boundary)
erosion1 = gBuffer(zone ,width = - (widthMax + 0.002) ,joinStyle="ROUND",capStyle = "ROUND")
erosion2 = gBuffer(zone ,width = - (widthMax - 0.002) ,joinStyle="ROUND",capStyle = "ROUND")
plot(erosion1)
```



```
plot(erosion2)
```



Smoothing all zones in zoning

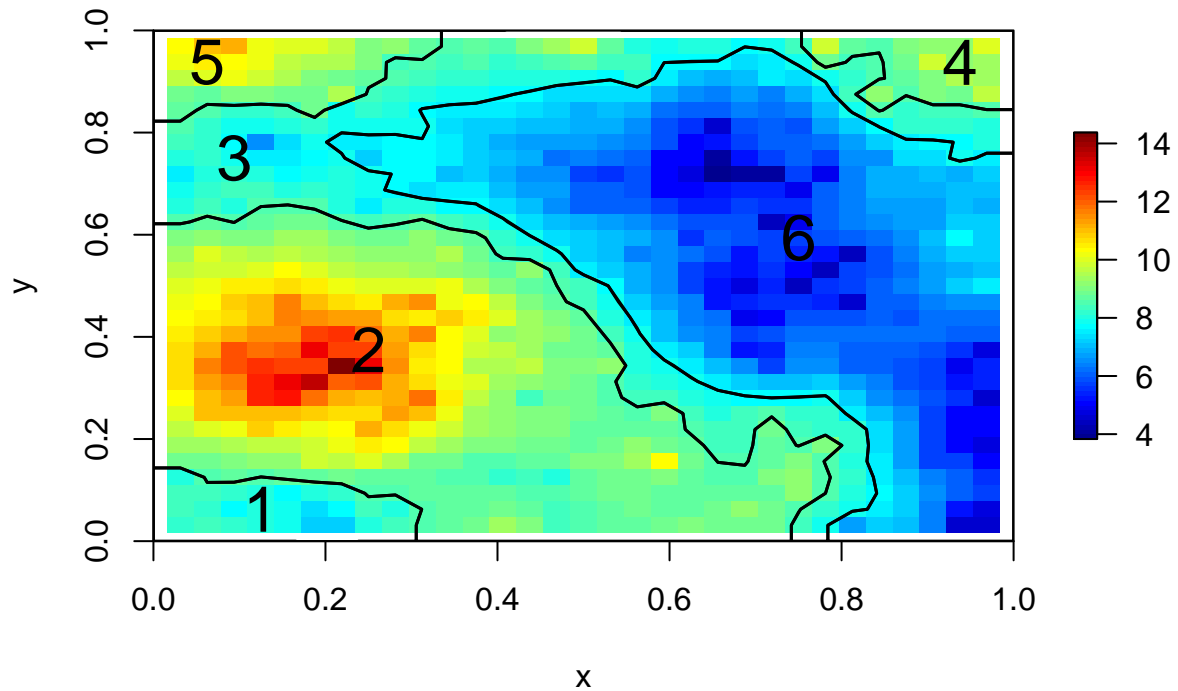
```
seed=1
map=genMap(DataObj=NULL,seed=seed,disp=FALSE,krig=2)

## [1] "DataObj=NULL, generating DataObj-seed= 1"
## [inverse distance weighted interpolation]

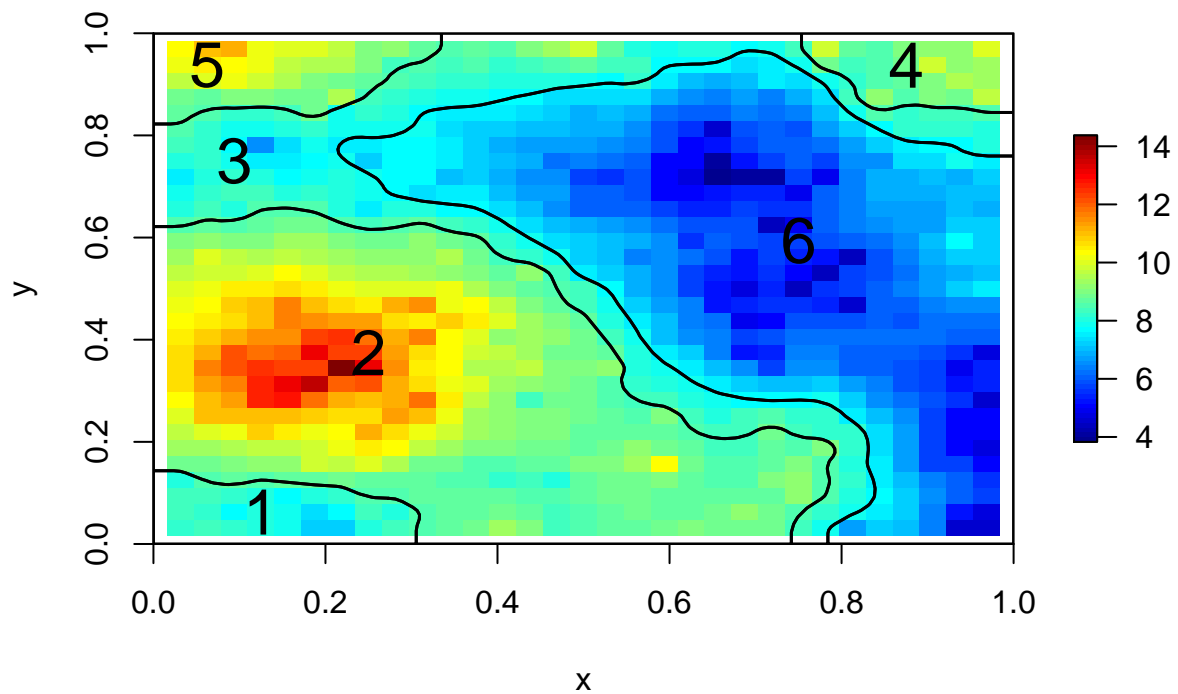
criti = correctionTree(qProb = c(0.4,0.6), map = map)
Z = criti$zk[[2]][[1]]$zonePolygone
newZ = smoothingMap(Z = Z, width = 0.05, map = map, disp = TRUE)

## [1] "nbZ.special<=2"
## [1] "nbZ.special = 2"
## [1] "smooth: 1"
## [1] "widthExt 0.05"
## [1] "widthInt 0.025"
## [1] "smooth: 4"
## [1] "widthExt 0.05"
## [1] "widthInt 0.025"
## [1] "smooth: 5"
## [1] "widthExt 0.05"
## [1] "widthInt 0.025"
## [1] "smooth: 6"
## [1] "widthExt 0.05"
## [1] "widthInt 0.025"
## [1] "union: 2 1"
## [1] "union: 3 4"
## [1] "union: 3 5"
## [1] "union: 3 6"
## [1] "before last zone: 2"
## [1] "widthExt 0.05"
## [1] "widthInt 0.025"
## [1] "remove: 1"
## [1] "last zone: 3"
## [1] "remove: 2"
## [1] "remove: 4"
## [1] "remove: 5"
## [1] "remove: 6"

plotM(map = map, Z = Z, lab = 1:length(Z))
```



```
plotM(map = map, Z = newZ, lab = 1:length(newZ))
```



ample of smoothing zoning done on real data

#Ex-

```
#data(yieldMapZ)
#plotM(map = map, Z = Z3, lab = 1:length(Z3))

#res = correctBoundaryMap(Zi = Z3, map = map)
#Z = res$Z
#newZ = smoothingMap(Z = Z, width = 0.05, map = map, disp = TRUE)
#plotM(map = map, Z = newZ, lab = 1:length(newZ))
```

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=C
##  [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] sp_1.2-4      ggplot2_2.2.1  rgeos_0.3-23  geozoning_1.0.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.11      compiler_3.4.0
##  [3] plyr_1.8.4        tools_3.4.0
##  [5] xts_0.9-7         digest_0.6.12
##  [7] gstat_1.1-5       evaluate_0.10.1
##  [9] tibble_1.3.1      gtable_0.2.0
## [11] lattice_0.20-35   rlang_0.1.1
## [13] yaml_2.1.14       spam_1.4-0
## [15] stringr_1.2.0     knitr_1.17
## [17] raster_2.5-8      RandomFieldsUtils_0.3.25
## [19] fields_8.15       maps_3.1.1
## [21] rprojroot_1.2     grid_3.4.0
## [23] spacetime_1.2-0   foreign_0.8-68
## [25] rmarkdown_1.6     deldir_0.1-14
## [27] magrittr_1.5      backports_1.1.0
## [29] scales_0.4.1      htmltools_0.3.6
## [31] intervals_0.15.1  RandomFields_3.1.50
## [33] maptools_0.9-2    colorspace_1.3-2
## [35] labeling_0.3       stringi_1.1.5
## [37] lazyeval_0.2.0    munsell_0.4.3
## [39] FNN_1.1           zoo_1.8-0
```