Qhull examples

David C. Sterratt

4th July 2022

This document presents examples of the `geometry` package functions which implement functions using the Qhull library.

1 Convex hulls in 2D

1.1 Calling `convhulln` with one argument

With one argument, convhulln returns the indices of the points of the convex hull.

```
> library(geometry)
> ps <-matrix(rnorm(30), , 2)
> ch <- convhulln(ps)
> head(ch)
```

```
     [,1] [,2]
[1,]   15   12
[2,]    8    1
[3,]    8   15
[4,]   14   12
[5,]   14    2
[6,]   11    1
```

1.2 Calling `convhulln` with `options`

We can supply Qhull options to `convhulln`; in this case it returns an object of class `convhulln` which is also a list. For example `FA` returns the generalised `area` and

`volume`. Confusingly in 2D the generalised area is the length of the perimeter, and the generalised volume is the area.

```
> ps <-matrix(rnorm(30), , 2)
> ch <- convhulln(ps, options="FA")
> print(ch$area)
```

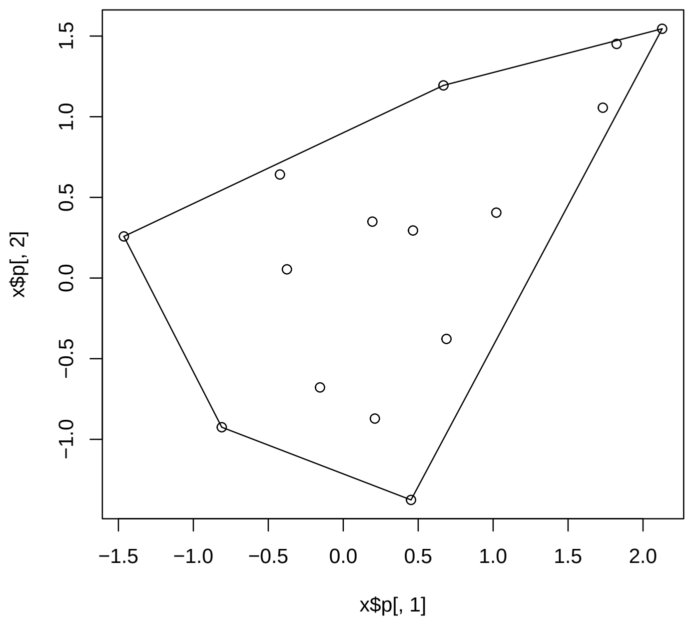
```
[1] 9.888223
```

```
> print(ch$vol)
```

```
[1] 5.074574
```

A `convhulln` object can also be plotted.

```
> plot(ch)
```



We can also find the normals to the "facets" of the convex hull:

```
> ch <- convhulln(ps, options="n")
> head(ch$normals)
```

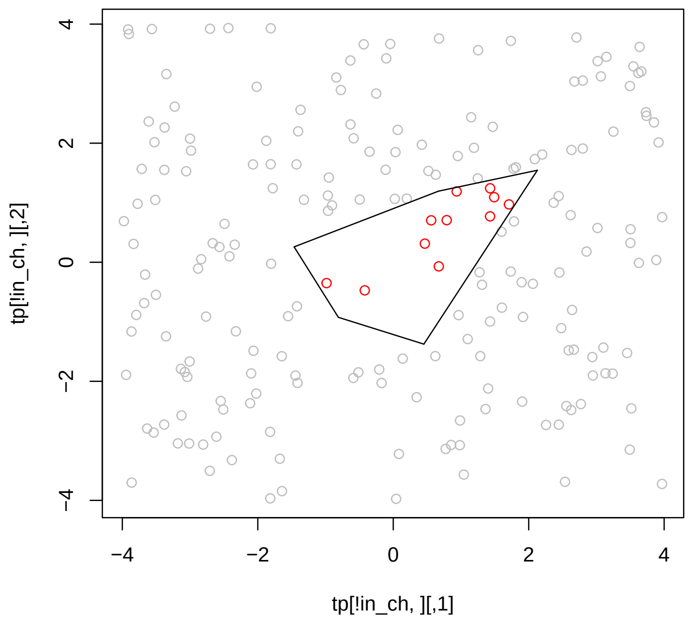
```
           [,1]       [,2]       [,3]
[1,]  0.8673940 -0.4976220 -1.0766129
[2,] -0.8754569 -0.4832963 -1.1565004
[3,] -0.3360776 -0.9418343 -1.1434805
[4,] -0.4020387  0.9156227 -0.8247258
[5,] -0.2338257  0.9722785 -1.0048426
```

Here the first two columns and the x and y direction of the normal, and the third column defines the position at which the face intersects that normal.

1.3 Testing if points are inside a convex hull with `inhulln`

The function `inhulln` can be used to test if points are inside a convex hull. Here the function `rbox` is a handy way to create points at random locations.

```
> tp <- rbox(n=200, D=2, B=4)
> in_ch <- inhulln(ch, tp)
> plot(tp[!in_ch,], col="gray")
> points(tp[in_ch,], col="red")
> plot(ch, add=TRUE)
```



2 Delaunay triangulation in 2D

2.1 Calling `delaunayn` with one argument

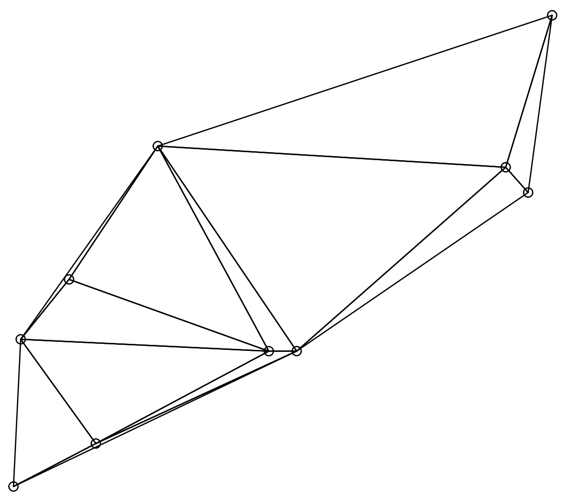
With one argument, a set of points, `delaunayn` returns the indices of the points at each vertex of each triangle in the triangulation.

```
> ps <- rbox(n=10, D=2)
> dt <- delaunayn(ps)
> head(dt)

     [,1] [,2] [,3]
[1,]    5    6    2
```

```
[2,]    5    9    2
[3,]    5    6    8
[4,]    5    9    8
[5,]    1    8    3
[6,]    1    7    3
```

```
> trimesh(dt, ps)
> points(ps)
```



2.2 Calling `delaunayn` with `options`

We can supply Qhull options to `delaunayn`; in this case it returns an object of class `delaunayn` which is also a list. For example `Fa` returns the generalised `area` of each triangle. In 2D the generalised area is the actual area; in 3D it would be the volume.

```
> dt2 <- delaunayn(ps, options="Fa")
> print(dt2$areas)
```

```
 [1] 0.001153860 0.013367272 0.001282930 0.060932360 0.005186326 0.077365072
 [7] 0.010641462 0.006477525 0.037343732 0.017458194 0.002890719 0.028252281
```

```
> dt2 <- delaunayn(ps, options="Fn")
> print(dt2$neighbours)
```

```
[[1]]
[1] -9  2 11

[[2]]
[1] -6  1 12

[[3]]
[1] -6  9 10

[[4]]
[1] -2  5  6

[[5]]
[1] -8  4  7

[[6]]
[1] 8 7 4

[[7]]
[1] -9  6  5

[[8]]
[1]  6 11  9

[[9]]
[1]  3  8 10

[[10]]
[1]  3 12  9

[[11]]
[1]  1  8 12

[[12]]
[1]  2 10 11
```