compGeometeR: an R package for computational geometry

Thomas R. Etherington
Manaaki Whenua – Landcare Research

O. Pascal Omondiagbe
Manaaki Whenua – Landcare Research

March 18, 2021

Abstract

Computational geometry algorithms and data structures are widely applied across numerous scientific domains, and there a variety of R packages that implement computational geometry functionality. However, these packages often work in specific numbers of dimensions, do not have directly compatible data structures, and include additional non-computational geometry functionality that can be domain specific. Our objective in developing the compGeometeR package is to implement in a generic and consistent framework the most commonly used combinatorial computational geometry algorithms so that they can be easily combined and integrated into domain specific scientific workflows. We briefly explain the discrete and digital combinatorial computational geometry algorithms available in compGeometeR, and identify priorities for future development.

Keywords: alpha complex, alpha shape, convex hull, convex layers, Delaunay triangulation, digital, discrete.

1 Introduction

Geometry is an ancient form of mathematics that enables a geometer to study the distance, shape, size, and position of objects in space (Gowers, 2003). Computational geometry is a more recent field of study that emerged alongside technological developments in computing and seeks to develop efficient algorithms and data structures for geometric objects. The two main areas of computational geometry are numerical computational geometry that considers continuous geometric objects (Kimmel, 2012) and combinatorial (or algorithmic) computational geometry that considers discrete geometric objects (Boissonnat and Yvinec, 1998; De Berg et al., 2008).

Our focus here is on combinatorial computational geometry, as these algorithms and data structures have been found to be widely applicable across numerous scientific domains (De Berg et al., 2008). Given the importance of combinatorial computational geometry it is no surprise that within the R (R Core Team, 2019) computing ecosystem there are a variety of options for applying various combinatorial computational geometry algorithms. However,

Table 1: Computational geometry software options in R. For each computational geometry function, the R software (and version) that can apply this algorithm is noted by reference to the number of dimensions in which the algorithm will work.

	compGeometeR (1.0.0)	R (3.5.3)	alphahull (2.2)	alphashape3d (1.3.1)	deldir (0.1-25)	geometry (0.4.5)	spatstat (1.63-3)	tripack (1.3-9.1)
Discrete algorithms								
Alpha complex	n							
Alpha shape			2	3				
Convex hull	n	2				n	2	2
Convex layers	n							
Delaunay triangulation	n		2		2	n	2	2
Voronoi diagram			2		2		2	2
Digital algorithms								
Alpha complex	n							
Alpha shape	n							
Convex hull	n							

these algorithms are currently spread across multiple somewhat incompatible packages that often work in specific numbers of dimensions (Table 1) and some include additional non-computational geometry functionality that can be domain specific. Also, these packages contain only one type of combinatorial computational geometry algorithm, those based on discrete objects such as points, lines, polygons. But other forms of combinatorial computational geometry such as digital geometry that studies the geometric properties of a grid (or lattice) of points (Rosenfeld and Melter, 1989; Klette and Rosenfeld, 2004) would be a useful addition to the R language.

Our objective in developing the compGeometeR (a computational geometer using R!) package is to produce a systematic implementation of some of the more commonly used discrete and digital combinatorial computational geometry algorithms so that they can be easily integrated into domain specific scientific workflows. Our hope is that by producing compGeometeR we can encourage the use of geometry in R for scientific study.

2 Algorithms

The algorithms of compGeometeR that are described in the following sections can be grouped into discrete algorithms and digital algorithms (Table 1), and the functions of compGeometeR that implement both kinds of algorithms have been specifically designed to easily interconnect and build upon one another (Figure 1). While some of the compGeometeR code is new, it is important to recognise that virtually all of the compGeometeR functions are dependent on computational geometry foundations provided by the Qhull C++ interface software (Barber

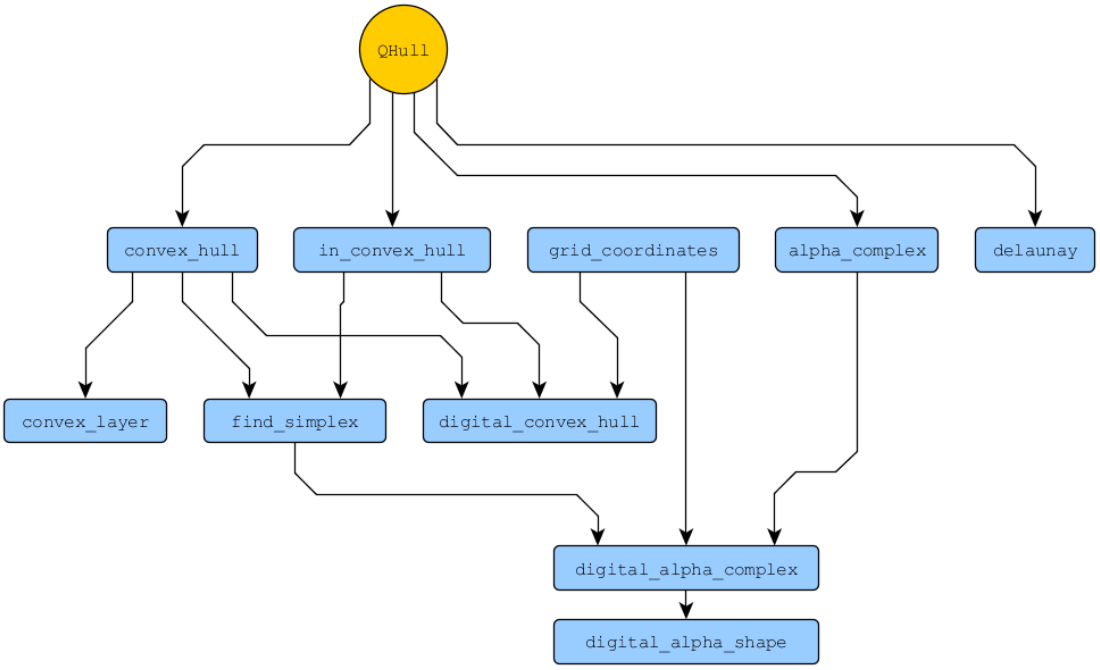

Figure 1: Connections and dependencies of the `compGeometeR` functions.

et al., 1996, `http://www.qhull.org/`) (Figure 1).

It is also important to note that while `compGeometeR`'s digital algorithms are unique to `R`, some of the discrete algorithms available in `compGeometeR` are also available in other `R` packages (Table 1). By documenting these similarities and differences we hope to help potential users to assess if `compGeometeR` is the best option for their needs as there may be other options that are more suitable.

2.1 Discrete algorithms

The discrete combinatorial computational geometry algorithms calculate the shapes and interactions between sets of discrete objects such as points, lines, circles, and polygons in 2-dimensional Euclidean space – and the hyperdimensional equivalents of these objects in n-dimensional Euclidean space. All of the discrete algorithms in `compGeometeR` take as an input a set of points P in n-dimensional Euclidean space $\mathbb{R}^n$.

A convex hull (Barber et al., 1996) defines the smallest subset of $\mathbb{R}^n$ that contains P and for which the subset is convex so any two points within the convex hull can be connected by a straight line also contained by the convex hull (Figure 2a). As an example of the simplicity of `compGeometeR` a convex hull can be generated and visualised with minimal code (Listing 1).

```
library(compGeometeR)
# Generate point data
set.seed(2) # to reproduce figure exactly
x = rgamma(n = 20, shape = 3, scale = 2)
y = rnorm(n = 20, mean = 10, sd = 2)
p = cbind(x, y)
# Create convex hull
ch = convex_hull(p)
```

```r
# Plot point data and convex hull
plot(x, y, yaxt="n", xaxt="n", xlab="", ylab = "", pch=16, cex=0.75)
polygon(ch$hull_vertices, col="orange", border="firebrick")
points(p, pch=16, cex=0.75)
```

Listing 1: Example `R` code to create a discrete convex hull with `compGeometeR`

Convex layers were first presented by Huber (1972) and Barnett (1976) who both gave unreferenced credit for this idea to Tukey. Convex layers are a nested sequence of convex hulls produced by repeating the process of constructing a convex hull for P and then removing the points forming the vertices of the convex hull from P before producing the next convex layer. The first convex layer is equivalent to the convex hull, with each successive convex layer representing ever smaller region of space (Figure 2b).

The Delaunay triangulation (Delaunay, 1934) produces a set of simplices (triangles in 2-dimensions or tetrahedrons in 3-dimensions) for which no point in P is inside the circumhypersphere (a circle in 2-dimensions or a sphere in 3-dimensions) of each simplex (Figure 2c).

The alpha complex (Edelsbrunner and Mücke, 1994) is a subset of the Delaunay triangulation that contains only those simplices for whose circumhypersphere radius is smaller than a specified α parameter value ranging $0 < \alpha < \infty$ (Figure 2d). The related alpha shape would be a polytope that combines all of the simplices in an alpha complex (Edelsbrunner and Mücke, 1994).

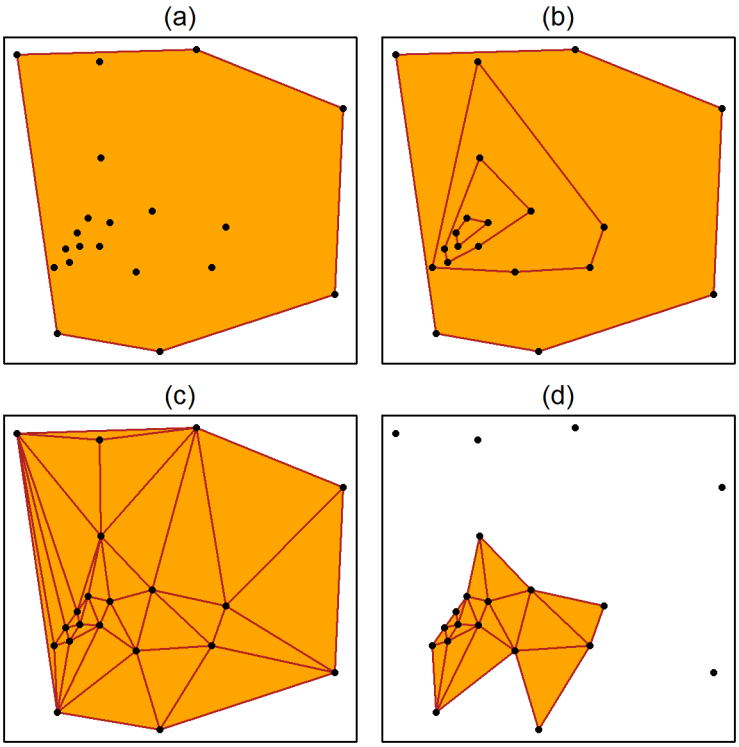

Figure 2: Two-dimensional examples of discrete geometry algorithms currently available in `compGeometeR`. (a) Convex hull, (b) convex layers, (c) Delaunay triangulation, and (d) alpha complex.

2.2 Digital algorithms

The digital combinatorial computational geometry algorithms calculate the geometric properties of a grid (or lattice) of points in Euclidean space, and usually involves the digitisation of discrete geometric objects (Rosenfeld and Melter, 1989). Digitisation occurs by representing Euclidean space $\mathbb{R}^n$ as a rectangular orthogonal grid $\mathbb{G}^n$. The elements of $\mathbb{G}^2$ are called pixels, and the elements of $\mathbb{G}^3$ are called voxels, and each element in $\mathbb{G}^n$ has a grid coordinate for its centre (Klette and Rosenfeld, 2004) and a value that denotes if the element belongs to a digitised geometric object, or when required, which part of the digitised geometric object. `compGeometeR` has implemented digital versions of the convex hull (Figure 3a), alpha complex (Figure 3b), and alpha shape (Figure 3c) that are unique functions amongst `R` software (Table 1).

The code required for digital versions of the discrete algorithms is only slightly more complex, and simply requires additional parameters to define the extent and resolution of $\mathbb{G}^n$. For example, the code required for a digital convex hull (Listing 2) does not differ much from that required for a discrete convex hull (Listing 1).

```
library(compGeometeR)
# Generate point data
set.seed(2) # to reproduce figure exactly
x = rgamma(n = 20, shape = 3, scale = 2)
y = rnorm(n = 20, mean = 10, sd = 2)
p = cbind(x, y)
# Create digital convex hull
d_ch = digital_convex_hull(p, mins=c(0,5), maxs=c(15,15), spacings = c
    (0.05,0.05))
# Plot point data and digital convex hull
image(x=d_ch[[3]][[1]], y=d_ch[[3]][[2]], z=d_ch[[1]],
      yaxt="n", xaxt="n", xlab="", ylab = "", col=c("white", "orange"))
points(p, pch=16, cex=0.75)
```

Listing 2: Example `R` code to create a digital convex hull with `compGeometeR`

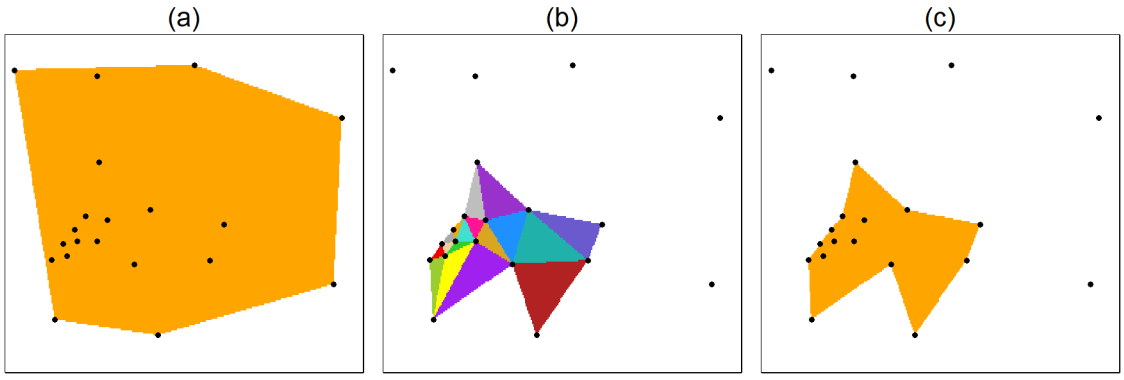

Figure 3: Two-dimensional examples of digital geometry algorithms currently available in `compGeometeR`. (a) Convex hull, (b) alpha complex, and (c) alpha shape.

3 Future work

`compGeometeR` is very much a work in progress, and while there is already sufficient functionality for `compGeometeR` to be useful in a wide range of computational sciences, there is some functionality that has been identified as being particularly useful for future development.

The Voronoi diagram (Voronoï, 1908; Okabe et al., 2000) partitions n-dimensional space into a set of polytopes for which each polytope delineates the region of n-dimensional space that is closest to each point in P. The Voronoi diagram would be an obvious addition for the discrete geometry algorithms given a 2-dimensional version has been implemented in other `R` geometry packages but no n-dimensional version is available (Table 1).

As well as expanding the number of discrete geometry algorithms, and implementing more discrete algorithms in digital form, there are other kinds of computational geometry algorithms that we think would be of particular value.

There are a variety of useful graph structures that can be produced based on geometric principles. For example, the Delaunay triangulation (Delaunay, 1934) can also be represented as a graph structure, and for which there are several useful subgraphs such as the Gabriel graph (Gabriel and Sokal, 1969), Urquhart graph (Urquhart, 1980), and relative neighbourhood graph (Toussaint, 1980).

In some contexts it will be important to recognise that there is uncertainty in the position of points in space and the parameters defining a given algorithm. In such situations it may be helpful to adopt fuzzy geometry (Rosenfeld, 1998) view in which, for example, memberships of points in space are not crisp being either 0 or 1, but rather are fuzzy on a scale from 0 to 1. Creating fuzzy versions of the `compGeometeR` algorithms could result in very useful functionality where quantifying uncertainty of any computational geometry is important. The existing digital geometric algorithms could be easily extended to fuzzy forms as all that is required is to assign a membership value to each pixel (Klette and Rosenfeld, 2004).

All the algorithms in `compGeometeR` work in Euclidean space, but computational geometry algorithms can also be usefully applied in spherical or hyperbolic space (Gowers, 2003). For example, the Fortran `STRIPACK` software (Renka, 1997) could be wrapped by `R` to provide functionality to compute Delaunay triangulations and Voronoi diagrams on the surface of a sphere.

4 Software availability

`compGeometeR` is open source software made available under a General Public License. Installation instructions can be found at the GitHub repository `https://github.com/manaakiwhenua/compGeometeR`.

We are also developing a cookbook of examples of `compGeometeR` use via the GitHub repository wiki `https://github.com/manaakiwhenua/compGeometeR/wiki`.

5 Acknowledgements

This research was funded by the New Zealand Ministry of Business, Innovation and Employment via the Beyond Myrtle Rust (#C09X1806) and Winning Against Wildings research programmes, with additional internal investment by Manaaki Whenua – Landcare Research

References

Barber CB, Dobkin DP, Huhdanpaa H (1996) The Quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software 22: 469–483

Barnett V (1976) The ordering of multivariate data. Journal of the Royal Statistical Society. Series A 139: 318–355

Boissonnat JD, Yvinec M (1998) Algorithmic Geometry. Cambridge University Press, Cambridge

De Berg M, Cheong O, van Kreveld M, Overmars M (2008) Computational Geometry, 3rd edition. Springer, Berlin

Delaunay B (1934) Sur la sphère vide. Bulletin de l'Académie des Sciences de l'URSS, Classe des Sciences Mathématiques et Naturelles 6: 793–800

Edelsbrunner H, Mücke EP (1994) Three-dimensional alpha shapes. ACM Transactions on Graphics 13: 43–72

Gabriel KR, Sokal RR (1969) A new statistical approach to geographic variation analysis. Systematic Biology 18: 259–278

Gowers T (2003) Mathematics: a very short introduction. Oxford University Press, New York

Huber PJ (1972) Robust statistics: a review. The Annals of Mathematical Statistics 43: 1041–1067

Kimmel R (2012) Numerical geometry of images: Theory, algorithms, and applications. Springer-Verlag, New York

Klette R, Rosenfeld A (2004) Digital Geometry. Morgan Kaufmann, San Francisco

Okabe A, Boots B, Sugihara K, Chiu SN (2000) Spatial Tessellations: concepts and applications of Voronoi diagrams, second edition edition. John Wiley & Sons, Chichester

R Core Team (2019) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria
URL `https://www.R-project.org/`

Renka RJ (1997) Algorithm 772: STRIPACK: Delaunay triangulation and Voronoi diagram on the surface of a sphere. ACM Transactions on Mathematical Software 23: 416–434

Rosenfeld A (1998) Fuzzy geometry an updated overview. Information Sciences 110: 127–133

Rosenfeld A, Melter RA (1989) Digital geometry. The Mathematical Intelligencer 11: 69–72

Toussaint GT (1980) The relative neighbourhood graph of a finite planar set. Pattern Recognition 12: 261–268

Urquhart RB (1980) Algorithms for computation of relative neighbourhood graph. Electronics Letters 16: 556–557

Voronoï G (1908) Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Premier mémoire. Sur quelques propriétés des formes quadratiques positives parfaites. Journal für die reine und angewandte Mathematik 133: 97–102