Appendix: Visualising How Non-linear Dimension Reduction Warps Your Data

Jayani P.G. Lakshika
Econometrics & Business Statistics, Monash University
and
Dianne Cook
Econometrics & Business Statistics, Monash University
and
Paul Harrison
MGBP, BDInstitute, Monash University
and
Michael Lydeamore
Econometrics & Business Statistics, Monash University
and
Thiyanga S. Talagala
Statistics, University of Sri Jayewardenepura

January 4, 2024

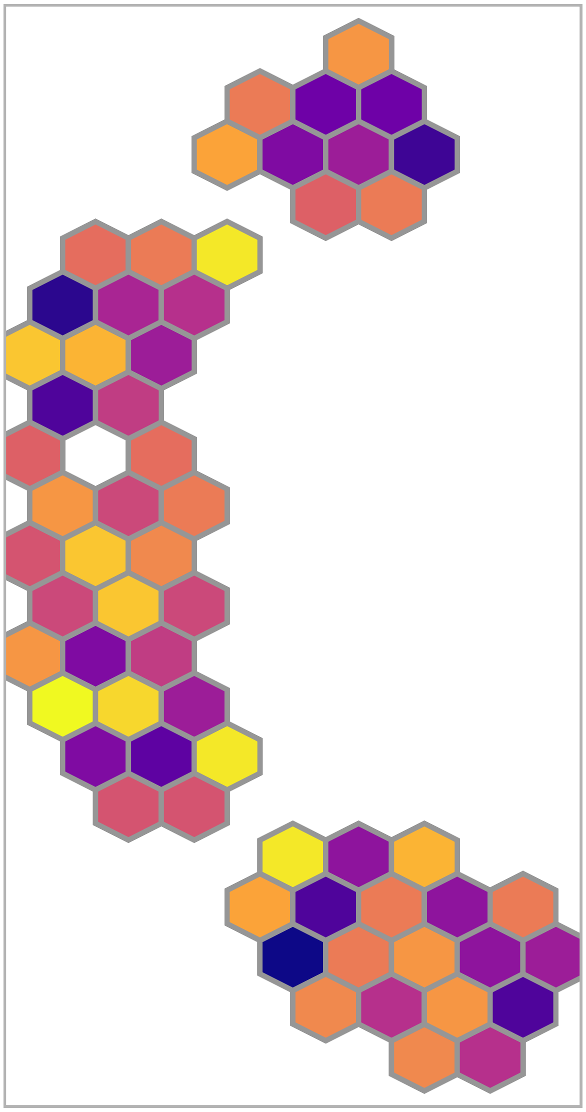
Abstract

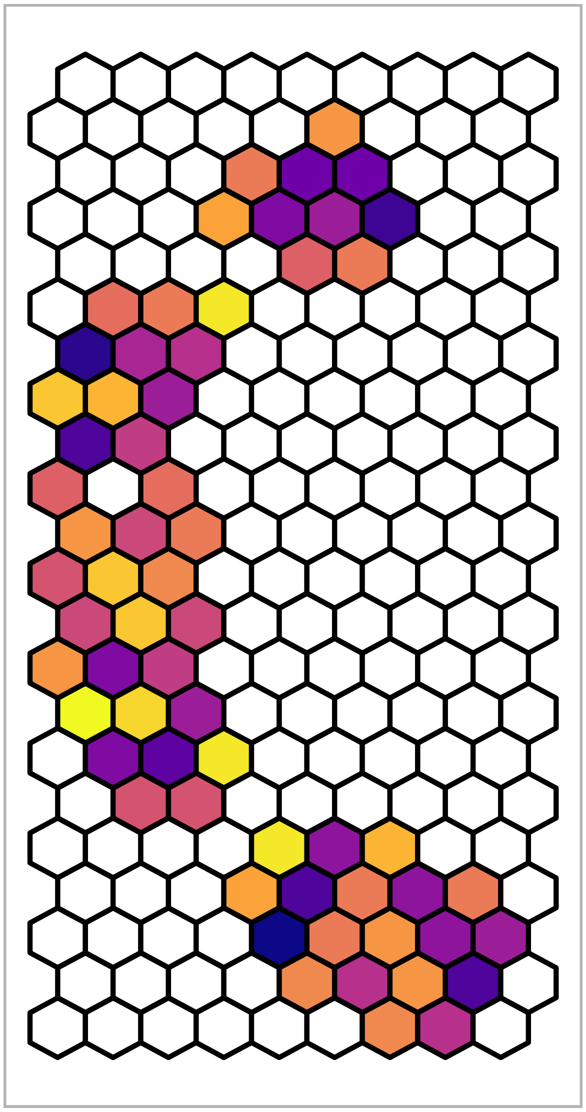
1 Total number of bins

The total number of bins represents the overall count of hexagonal bins within the hexagonal grid. This count is determined by multiplying the number of bins along the x-axis (b_1) with the number of bins along the y-axis (b_2), according to the formula:

$$b = b_1 \times b_2 \tag{1}$$

Here, b denotes the total number of bins. By adjusting the parameter b_1, we have control over the total number of bins (b). Hence, fine-tuning b_1 enables us to customize and optimize the total bin count based on the desired configuration along the x-axis.





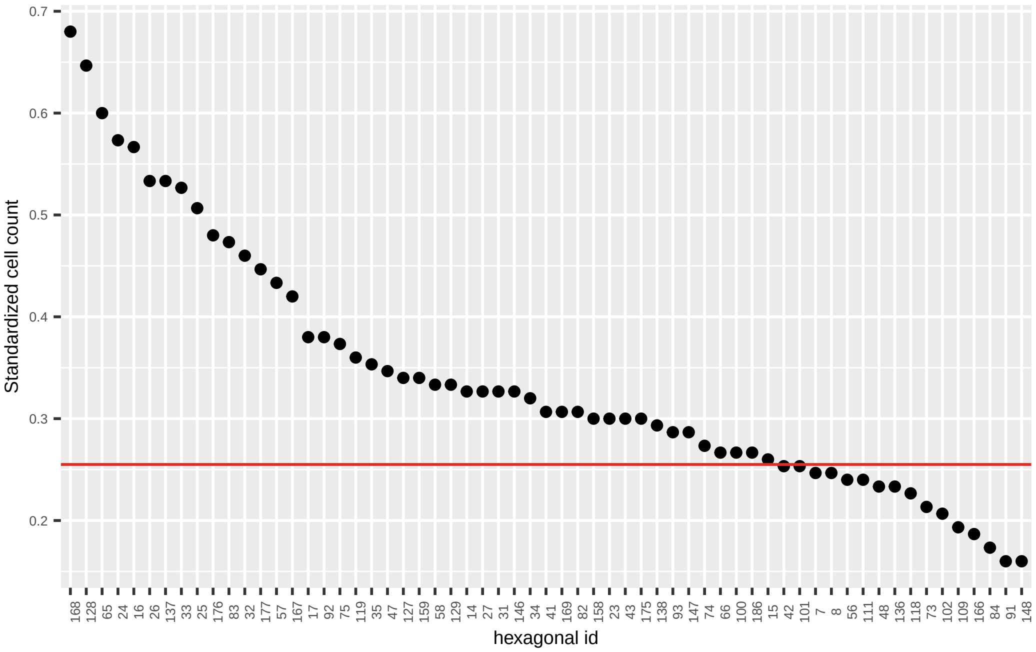
2 Benchmark value to remove low-density hexagons

In the step of addressing low-density hexagons, which can arise from sparsely represented data in certain regions, we employ a systematic strategy. The goal is to ensure a more comprehensive coverage of the data by removing hexagons with low data density. To achieve this, we initiate the process by identifying, for each hex bin, the six nearest hex bins based on an equal 2D distance metric. Following this, we calculate the mean density (see Equation 3), as outlined in the equation:

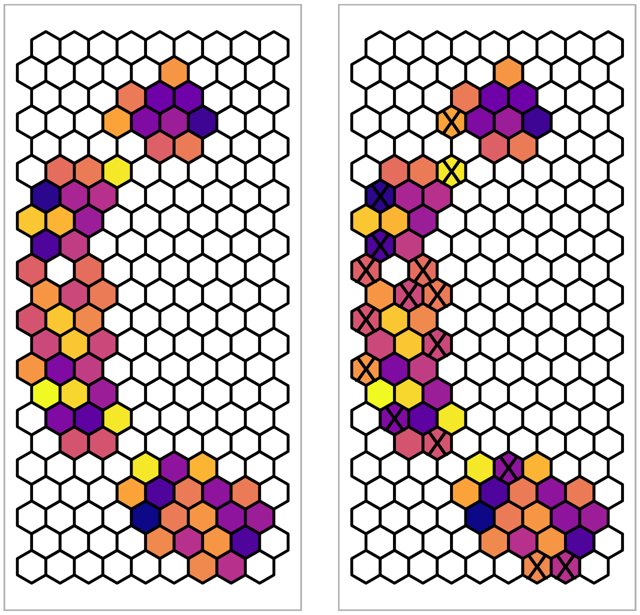
$$\text{standard count} = \frac{\text{count}}{\text{max count}} \tag{2}$$

$$\text{mean density} = \frac{\text{standard count}}{6} \tag{3}$$

The standard count is derived from the number of observations in the hex bins (see Equation 2). Next, we examine the distribution of mean densities across all hex bins and designate the first quartile as the benchmark value for removing low-density hexagons. Finally, hex bins with mean densities below this benchmark value are removed from consideration. This meticulous procedure ensures the elimination of regions with inadequate data density, allowing the focus to shift to areas with more significant data representation. The result is the preservation of the overall structure of the data in the low-dimensional space, as illustrated in **?@fig-bintorm** (c).

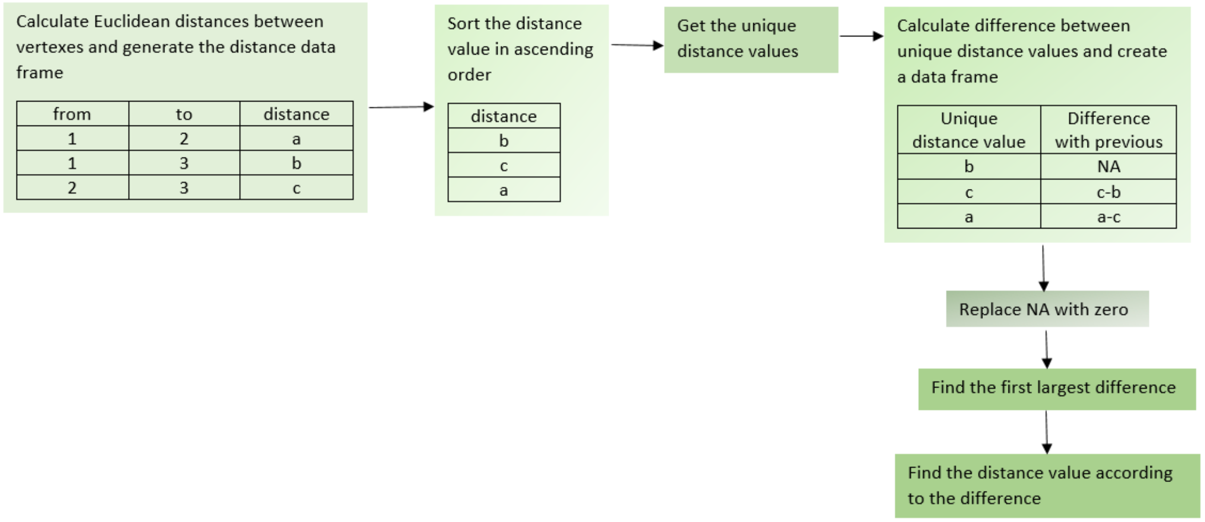


```
Joining with `by = join_by(hexID)`
```



3 Benchmark value to remove long edges

In this step, we aim to create a smoother surface in the low-dimensional space by removing long edges from the triangular mesh. This process helps eliminate outliers and noise while preserving important local relationships and intricate structures within the data. To achieve this, distances between vertices are sorted, and unique distance values are extracted. A data frame is created to calculate the differences between consecutive distance values, and the largest difference serves as a threshold to identify long edges. By removing edges that exceed this threshold, the algorithm refines the triangular mesh (see **?@fig-traingularmeshgr**).



Calculate Euclidean distances between vertexes and generate the distance data frame

from	to	distance
1	2	a
1	3	b
2	3	c

Sort the distance value in ascending order

distance
b
c
a

Get the unique distance values

Calculate difference between unique distance values and create a data frame

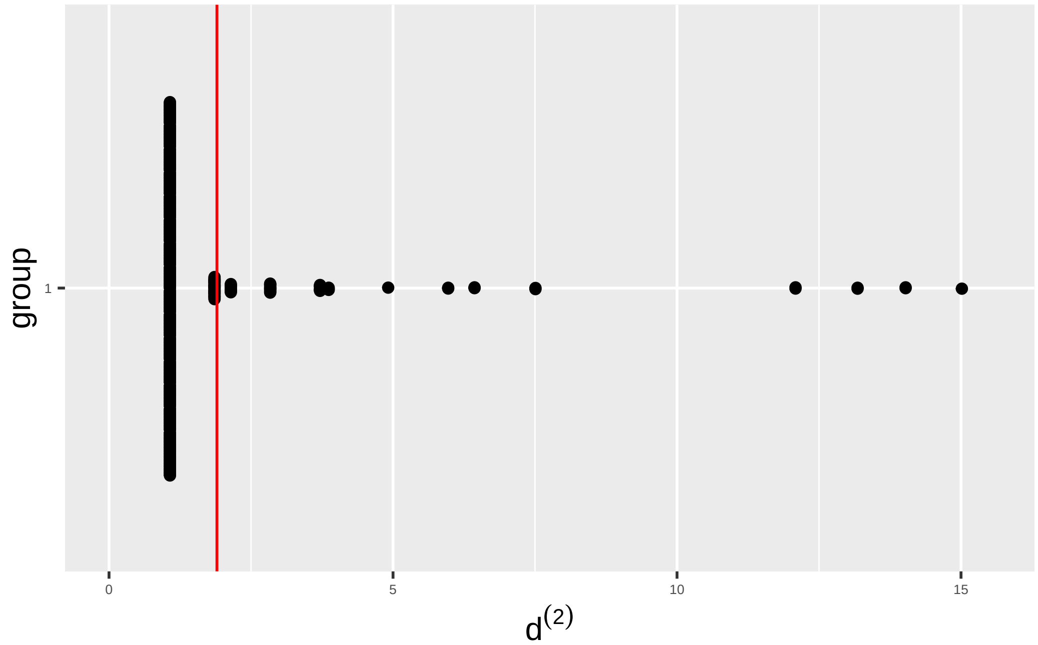
Unique distance value	Difference with previous
b	NA
c	c-b
a	a-c

Replace NA with zero

Find the first largest difference

Find the distance value according to the difference

Figure 1: A flow diagram detailing the steps taken to find the benchmark value to remove long edges.


group
1
0
5
10
15
d$^{(2)}$