cardinalR: Generating interesting high-dimensional data structures

by Jayani P. Gamage, Dianne Cook, Paul Harrison, Michael Lydeamore, and Thiyanga S. Talagala

Abstract A high-dimensional dataset is where each observation is described by many features, or dimensions. Such a dataset might contain various types of structures that have complex geometric properties, such as nonlinear manifolds, clusters, or sparse distributions. We can generate data containing a variety of structures using mathematical functions and statistical distributions. Sampling from a multivariate normal distribution will generate data in an elliptical shape. Using a trigonometric function we can generate a spiral. A torus function can create a donut shape. High-dimensional data structures are useful for testing, validating, and improving algorithms used in dimensionality reduction, clustering, machine learning, and visualization. Their controlled complexity allows researchers to understand challenges posed in data analysis and helps to develop robust analytical methods across diverse scientific fields like bioinformatics, machine learning, and forensic science. Functions to generate a large variety of structures in high dimensions are organized into the R package `cardinalR`, along with some already generated examples.

1 Introduction

Generating synthetic datasets with clearly defined geometric properties is essential for evaluating and benchmarking algorithms in various fields, such as machine learning, data mining, and computational biology. Researchers often need to generate data with specific dimensions, noise characteristics, and complex underlying structures to test the performance and robustness of their methods.

There are numerous packages available in R for generating synthetic data, each designed with unique characteristics and focus areas. For example, `geozoo` (Schloerke (2016)) offers a large collection of geometric objects, allowing users to create and analyze specific shapes, primarily in lower-dimensional spaces. Another useful package is `snedata` (Melville (2025)), which provides tools for generating simplified datasets useful for evaluating dimensionality reduction techniques like tSNE, often focusing on understanding and evaluating low-dimensional embeddings of complex data structures. Additionally, `mlbench` (Leisch and Dimitriadou (2024)) includes a collection of well-known benchmark datasets commonly associated with established classification or regression challenges. In the field of single-cell omics, `splatter` (Zappia et al. (2017)) is particularly simulate complex biological data, effectively capturing nuances such as batch effects and differential expression.

There is a valuable opportunity to improve the generation of high-dimensional data structures by integrating geometric principles with advanced noise control and customizable clustering. The `geozoo` package provides a strong foundation but could be enhanced to support high-dimensional extensions with controlled noise and user-defined parameters for clustering. Similarly, while `snedata` focuses on abstract datasets for dimensionality reduction, adding features for generating high-dimensional data from geometric layouts would enhance its usability. The `mlbench` package could also benefit from allowing users to create datasets with specific geometric structures and noise profiles. Additionally, although `splatter` specializes in biological data simulation, it could be expanded to offer a broader framework for generating diverse geometric structures across dimensions, enabling detailed control over noise and clustering. Addressing these areas could lead to more robust high-dimensional data generation tools.

To address this gap, this paper introduces the `cardinalR` R package. This package provides a collection of functions designed to generate customizable data structures in any number of dimensions, starting from basic geometric shapes. `cardinalR` offers important functionalities that extend beyond the capabilities of existing tools, allowing users to: (i) construct high-dimensional datasets based on geometric shapes, including the option to enhance dimensionality by adding controlled noise dimensions; (ii) introduce adjustable levels of background noise to these structures; and (iii) create complex clustered data arrangements by using fundamental geometric forms, while maintaining their positions, scales, orientations, and sample sizes in arbitrary dimensional spaces. By providing these integrated features, `cardinalR` aims to provide researchers to generate more explainable and challenging synthetic datasets focused to the specific needs of evaluating algorithms in high-dimensions. This bridges the gap between geometric foundations and the flexible generation of complex synthetic data.

The paper is organized as follows. In next section, introduces the implementation of `cardinalR` package on CRAN and GitHub, including demonstration of the package's key functions. We illustrate how a clustering data structure affect the Non-linear dimension reductions in **Example** section. Finally, we give a brief conclusion of the paper and discuss potential opportunities for use of our data

Table 1: cardinalR data sets

data	explanation
mobius_clust_data	
mobius_clust_tsne_param1	
mobius_clust_tsne_param2	
mobius_clust_tsne_param3	
mobius_clust_umap_param1	
mobius_clust_umap_param2	
mobius_clust_umap_param3	

collection.

2 Implementation

Installation

The package can be installed from CRAN using

```
install.packages("cardinalR")
```

and the development version can be installed from GitHub

```
devtools::install_github("JayaniLakshika/cardinalR")
```

Web site

More documentation of the package can be found at the web site https://jayanilakshika.github.io/cardinalR/.

Data sets

The cardinalR package comes with several data sets that load with the package. These are described in Table 1.

```
datasets_tb |>
 kable(caption = "cardinalR data sets", format="latex", col.names = c("data", "explanation"), booktabs = T) |>
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

Functions

Main function

```
gen_multicluster(
  n = c(200, 300, 500), p = 4, k = 3,
  loc = matrix(c(
    0, 0, 0, 0,
    5, 9, 0, 0,
    3, 4, 10, 7
  ), nrow = 4, byrow = TRUE),
  dim_weights = dim4_weights,
  scale = c(3, 1, 2),
  shape = c("gaussian", "bluntedcorn", "unifcube"),
  rotation = rotations_4d,
  is_bkg = FALSE
)
```

Table 2: cardinalR branching data generation functions

Function	Explanation
gen_expbranches	
gen_linearbranches	
gen_curvybranches	
gen_orglinearbranches	
gen_orgcurvybranches	

Table 3: cardinalR branching data generation functions

Function	Explanation
gen_expbranches	
gen_linearbranches	
gen_curvybranches	
gen_orglinearbranches	
gen_orgcurvybranches	

Branching

```
branch_tb |>
 kable(caption = "cardinalR branching data generation functions", format="latex", col.names = c("Function", "Exp
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

Cone

Cube

```
cube_tb |>
 kable(caption = "cardinalR cube data generation functions", format="latex", col.names = c("Function", "Explanat
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

Gaussian

Linear

Mobius

Polynomial

```
polynomial_tb |>
 kable(caption = "cardinalR polynomial data generation functions", format="latex", col.names = c("Function", "Ex
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

Table 4: cardinalR cube data generation functions

Function	Explanation
gen_gridcube	
gen_unifcube	
gen_cubehole	

Table 5: cardinalR cube data generation functions

Function	Explanation
gen_gridcube	
gen_unifcube	
gen_cubehole	

Table 6: cardinalR polynomial data generation functions

Function	Explanation
gen_quadratic	
gen_cubic	

Pyramid

```
pyramid_tb |>
 kable(caption = "cardinalR pyramid data generation functions", format="latex", col.names = c("Function", "Expla
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

S-curve

Sphere

```
sphere_tb |>
 kable(caption = "cardinalR sphere data generation functions", format="latex", col.names = c("Function", "Explan
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

Swiss Roll

To generalize the Swiss roll structure to arbitrary dimensions, we introduce a function `generate_swiss_roll(n, p)`, which constructs a high-dimensional version of the classic 3D Swiss roll while preserving its core characteristics.

The function generates `n` points in a `p`-dimensional space, where the first two dimensions (`X_1`, `X_2`) define the primary Swiss roll shape using a parametric equation:

$$X_1 = t\cos(t), \quad X_2 = t\sin(t), \quad \text{where } t \sim U(0, 3\pi)$$

The third dimension (`X_3`) introduces variation perpendicular to the roll, sampled uniformly from $[-1, 1]$. Additional dimensions (`X_4` to `X_p`) extend the data structure by applying a **sinusoidal transformation** of the parameter `t`, ensuring continuity in higher-dimensional spaces:

$$X_i = \frac{\sin(it)}{i}, \quad \text{for } i \geq 4.$$

This transformation ensures a gradual decay in variance across dimensions, mimicking real-world high-dimensional structures where later dimensions often capture subtler variations.

Table 7: cardinalR polynomial data generation functions

Function	Explanation
gen_quadratic	
gen_cubic	

Table 8: cardinalR pyramid data generation functions

Function	Explanation
gen_pyr	
gen_pyrrect	
gen_pyrtri	
gen_pyrstar	
gen_pyrholes	

Table 9: cardinalR pyramid data generation functions

Function	Explanation
gen_pyr	
gen_pyrrect	
gen_pyrtri	
gen_pyrstar	
gen_pyrholes	

Table 10: cardinalR S-curve data generation functions

Function	Explanation
gen_scurve	
gen_scurvehole	

Table 11: cardinalR sphere data generation functions

Function	Explanation
gen_circle	
gen_curvycycle	
gen_unifsphere	
gen_gridedsphere	
gen_clusteredspheres	

Table 12: cardinalR sphere data generation functions

Function	Explanation
gen_circle	
gen_curvycycle	
gen_unifsphere	
gen_gridedsphere	
gen_clusteredspheres	

Table 13: cardinalR trigonometric data generation functions

Function	Explanation
gen_crescent	
gen_curvycylinder	
gen_sphericalspiral	
gen_helicalspiral	
gen_conicspiral	
gen_nonlinear	

Table 14: cardinalR trigonometric data generation functions

Function	Explanation
gen_crescent	
gen_curvycylinder	
gen_sphericalspiral	
gen_helicalspiral	
gen_conicspiral	
gen_nonlinear	

Trigonometric

```
trigonometric_tb |>
 kable(caption = "cardinalR trigonometric data generation functions", format="latex", col.names = c("Function",
  column_spec(1, width = "4cm") |>
  column_spec(2, width = "8cm")
```

Odd shapes

Additional functions

3 Example

Add one or two datasets and evaluate how it will be useful. . . . (one with NLDR, one with clustering)

4 Discussion

5 Code

The code is available at https://github.com/JayaniLakshika/cardinalR, and source material for this paper is available at https://github.com/JayaniLakshika/paper-cardinalR.

6 Acknowledgements

This article is created using **knitr** (Xie, 2015) and **rmarkdown** (Xie et al., 2018) in R with the `rjtools::rjournal_article` template.

Bibliography

F. Leisch and E. Dimitriadou. *mlbench: Machine Learning Benchmark Problems*, 2024. URL https://CRAN.R-project.org/package=mlbench. R package version 2.1-6. [p1]

J. Melville. *snedata: SNE Simulation Dataset Functions*, 2025. URL https://github.com/jlmelville/snedata. R package version 0.0.0.9001, commit beebcf91c365bf5006be08fb614585b4659c05c5. [p1]

B. Schloerke. *geozoo: Zoo of Geometric Objects*, 2016. URL https://CRAN.R-project.org/package=geozoo. R package version 0.5.1. [p1]

Y. Xie. *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition, 2015. URL https://yihui.name/knitr/. ISBN 978-1498716963. [p6]

Y. Xie, J. Allaire, and G. Grolemund. *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida, 2018. URL https://bookdown.org/yihui/rmarkdown. ISBN 978-1138359338. [p6]

L. Zappia, B. Phipson, and A. Oshlack. Splatter: simulation of single-cell rna sequencing data. *Genome Biology*, 2017. doi: 10.1186/s13059-017-1305-0. URL http://dx.doi.org/10.1186/s13059-017-1305-0. [p1]

Jayani P. Gamage
Monash University
Department of Econometrics and Business Statistics, VIC 3800 Australia
https://jayanilakshika.netlify.app/
ORCiD: 0000-0002-6265-6481
jayani.piyadigamage@monash.edu

Dianne Cook
Monash University
Department of Econometrics and Business Statistics, VIC 3800 Australia
http://www.dicook.org/
ORCiD: 0000-0002-3813-7155
dicook@monash.edu

Paul Harrison
Monash University
MGBP, BDInstitute, VIC 3800 Australia
ORCiD: 0000-0002-3980-268X
paul.harrison@monash.edu

Michael Lydeamore
Monash University
Department of Econometrics and Business Statistics, VIC 3800 Australia
ORCiD: 0000-0001-6515-827X
michael.lydeamore@monash.edu

Thiyanga S. Talagala
University of Sri Jayewardenepura
Department of Statistics, Gangodawila, Nugegoda 10100 Sri Lanka
https://thiyanga.netlify.app/
ORCiD: 0000-0002-0656-9789
ttalagala@sjp.ac.lk