SoftwareX 24 (2023) 101514

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

sphstat: A Python package for inferential statistics on vectorial data on the unit sphere

Hüseyin Hacıhabiboğlu

Graduate School of Informatics, Middle East Technical University (METU), Ankara, TR 06800, Turkiye

ARTICLE INFO

Keywords:
Spherical data
Inferential statistics
Hypothesis testing
Goodness-of-fit
Spherical regression

ABSTRACT

Data that resides on the surface of a 2-sphere is common in various scientific fields, including physics, earth sciences, astronomy, and psychoacoustics. While some tools and packages exist for performing inferential statistical tests on such data and model fitting, there is currently no comprehensive open-source Python package that implements these tests. `sphstat` aims to fill this gap by providing an open-source Python package that implements spherical inferential tests and some model-fitting algorithms as catalogued in the authoritative reference by Fisher et al. (1993). Due to the lack of a similar open-source Python package, `sphstat` has the potential to be widely used in scientific and technical fields where data on the 2-sphere emerges.

Code metadata

Current code version	1.0.5
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-22-00453
Permanent link to Reproducible Capsule	https://bit.ly/sphstat
Legal Code License	MIT License
Code versioning system used	Git
Software code languages, tools, and services used	Python (≥3.8)
Compilation requirements, operating environments & dependencies	Tested on MacOS 12.0.1, Requirements: `numpy`, `scipy`, `matplotlib`, `openpyxl`, `sympy`, `pandas`, `setuptools`
Link to developer documentation/manual	http://sphstat.readthedocs.io
Support email for questions	huseyin@hacihabiboglu.org

1. Introduction

Data on the unit sphere appear in many branches of science. For example, positions of solar flares on the sun [1], eye tracking data [2], data in earth sciences [3,4], data from neuroimaging studies [5], data obtained for terrestrial laser scanning based mapping [6] or directional responses elicited from subjects in a 3D egocentric localisation experiment [7,8] can be represented as data on the unit sphere.

Despite the ubiquitous nature of spherical data, inferential analysis thereof is a topic of ongoing research [9]. Hypothesis testing on uni- or higher-dimensional data has been developing since at least the early 18th century [10] and more notably from 20th-century onwards [11]. There are well-established open-source software such as JASP [12] and open-source Python libraries such as SciPy [13] and Pingouin [14] that implement a variety of statistical tests on linear data.

There exist open source software packages such as the R packages `Directional` [15] and `rcosmo` [16] for the analysis of spherical data. However, despite developments that have been going on for quite some time [17,18] inferential statistics and modelling of data on the unit sphere are not adequately catered for by any publicly available, open-source Python package. The motivation behind the development of `sphstat` is based on a real need that arose while working with directional data in the spatial auditory perception domain. The present author, along with two other researchers, has collected subjective localisation data for two different loudspeaker-based 3D audio reproduction systems with the aim to understand whether the differences in median directions pointed at by the subjects for a sound source that is rendered nominally at the same direction by the two methods were statistically significant [19]. The development of `sphstat` was started to facilitate the required directional analyses.

`sphstat` comprises implementations in Python of the descriptive statistics calculations, inferential statistics tests and modelling tools as

E-mail address: hhuseyin@metu.edu.tr.

https://doi.org/10.1016/j.softx.2023.101514
Received 30 December 2022; Accepted 22 August 2023

catalogued by Fisher, Lewis and Embleton [20], as well as some basic plotting utilities.

2. Implementation

`sphstat` is written in Python 3 [21]. Array-based calculations are written using NumPy [22] and several statistics-related functions as well as special function are imported from SciPy [13]. `sphstat` uses SymPy [23] for the exact solutions of some defining equations, `pandas` [24] and `openpyxl` [25] for importing and accessing data, and `matplotlib` [26] for visualisation. `sphstat` is organised into several modules that align with their primary use cases (e.g. distributions, descriptive statistics, single sample tests, multiple sample tests, modelling, plotting and utility functions). `sphstat` documentation was built with Sphinx [27] and is available at https://sphstats.readthedocs.io. `sphstat` is released under the MIT License.

3. Features

`sphstat` comprises several modules that implement the functionality required of a statistical library. These are the descriptive statistics module (`sphstat.descriptives`), distributions module (`sphstat.distributions`), single sample tests module (`sphstat.singlesample`), multiple sample tests module (`sphstat.twosample`), modelling module (`sphstat.modelling`), plotting module (`sphstat.plotting`), and utility functions (`sphstat.utils`).

`sphstat.descriptives` module implements functionality to extract summary statistics from a given data including the directional cosines, resultant vector, resultant length, mean direction and the mean resultant length. Some of these quantities are used in hypothesis testing. There are also other functions to estimate the median of a sample, and to calculate the shape parameters of the sample.

`sphstat.distributions` module implements uniform spherical, Fisher, Fisher-Bingham, Kent, and Watson distributions and can be used to generate samples of random unit vectors from these distributions. The module relies on the `numpy.random` module to generate uniform and normally distributed numbers used for calculating the samples.

`sphstat.singlesample` module implements several estimation and hypothesis test methods to be applied onto a sample of observations on the unit sphere. Specifically the module includes functions that implement:

1. Tests for assessing whether the sample comes from (i) a spherical uniform distribution, (ii) a rotationally symmetric distribution, (iii) a Fisher distribution and (iv) a Kent distribution as opposed to a Fisher distribution
2. Tests that are akin to single sample t-tests for testing the distribution mean or median against a given mean or median
3. Algorithms for estimating the mean, median, confidence cone, as well as parameters for the sample under the assumptions that the sample is drawn from a general axisymmetric, Fisher or Kent distribution.
4. Algorithms to estimate parameters of bimodal distributions under the assumption that the sample is drawn from a Wood distribution.

`sphstat.twosample` module implements tests for two or more samples of unimodal vectorial data. Specifically, the following functionalities are implemented:

1. Tests for assessing whether two or more samples have the same mean or median with or without the assumption that the samples come from a Fisher distribution
2. Algorithms for estimating the pooled mean and median for two or more samples with or without the assumption that the samples come from a Fisher distribution
3. Functions for calculating the common mean, median or concentration parameter for multiple samples

`sphstat.modelling` module implements algorithms for the correlation, regression and spatiotemporal analysis of vectorial data on the sphere. Specifically, the module implements the following functionality:

1. Calculation of the correlation of two random unit vectors on the sphere given two samples
2. Calculation of the correlation of a variable with unit vectors on the sphere
3. Regression of a random unit vector on a circular variable
4. Time series analysis of vectors on the unit sphere

4. Omissions and limitations

Some of the tests or methods described in [20] were not included in this first major version of `sphstat`. More specifically, the estimation of the distribution parameters for small and great circle distributions, and methods for distributions of unidirected lines are not included in the present version. It is planned that future versions of the package will be extended to include tests and methods for axial and girdle data alongside vectorial data.

Some of the tests described in [20] extensively rely on data presented as nomograms or in extensive tables, which are to be used under certain conditions (e.g. when the sample size or the concentration parameters is small). While it might have been possible to integrate these data into the package, a deliberate choice was made not to include them in this version as these typically represent edge cases with limited applicability. Still, most practical cases (e.g. for larger sample sizes) are already covered by approximate methods that are implemented as part of `sphstat`. Extension of the package for these edge cases is planned for future versions.

Plotting capabilities, while sufficient for most practical purposes, are rather limited since this is not the main purpose of the package. However, future versions are planned to incorporate more elaborate, possibly interactive plotting methods.

5. Illustrative examples

Several examples demonstrating the usage of `sphstat` are presented in this section. In the examples below, we use data reproduced in table form in [20]. We also provide relevant references to the original papers.

5.1. Example 1: Importing, generating and plotting samples

`sphstat` uses the polar coordinates with the colatitute (i.e. θ) and longitude (ϕ) angles given in the fundamental ranges of $0 \leq \theta < 2\pi$ and $0 \leq \phi \leq \pi$, respectively. While `sphstat` internally uses polar coordinates, data in other commonly used representations such as Declination/Inclination and Latitude/Longitude can also be used after an appropriate conversion is carried out. Some domain-specific coordinates such as Plunge/Azimuth and other geographical or astronomical coordinates are not covered.

In its present version, `readsample` function in `sphstat.utils` can only import data in Excel (i.e. `.xlsx`) files. This provides, as opposed to simpler formats such as comma separated values (i.e. `.csv`) files, the ability to store multiple samples in different worksheets of the same file. Each row will contain a single observation with the first column containing colatitute, inclination, or latitude angles, and the second column containing longitude or declination angles, respectively. The data can be either in degrees or in radians which must be specified while the sample is being imported. Individual samples can be plotted using `plotdata` either in Mollweide or in Lambert projections. Multiple samples can be overlaid using `plotdatalist` in the same

Listing 1
Basic import, random sample generation and plotting operations.

```
import numpy as np

# Import the sample stored in file
sample1 = readsample('/PATH/TO/data.xlsx', typ='rad')
# Data is in Declination/Inclination coordinates. Convert to polar
samplerad1 = convert(sample1, 'decinc')
# Create a two samples containing 100 random vectors from Fisher distribution
# with concentration parameters of 10 and 5.
sample2 = fisher(100, np.pi/3, 3 * np.pi/2, 20)
sample3 = fisher(100, np.pi/3, np.pi/2, 5)
# Convert from Cartesian to polar
samplerad2 = carttopolar(sample)
samplerad3 = carttopolar(sample)
# Calculate the descriptive statistics for the sample and print
r = resultants(sample2)
prettyprintdict(r)

Directional Cosines:  [-0.00956974 -0.85529769  0.51804853]
Resultant Vector:  [ -0.86648713 -77.44245481  46.90641634]
Resultant Length:  90.54444489249717
Mean Direction:  (1.026228443399094, -1.5819846469608945)
Mean Resultant Length:  0.9054444489249717

# Rotate the imported sample by -pi in latitude
samplecart1 = polartocart(sample1rad)
samprot = rotatesample(sample1cart, 0, -np.pi, 0)
samplerad1 = carttopolar(samprot)

# Make a list of samples
samplelist = [samplerad1, samplerad2, samplerad3]
# Plot the samples in Mollweide and Lambert projections
plotdatalist(samplelist, labels=['Data', 'Kappa=20', 'Kappa=5'], proj='mollweide',
                 mflag = True)
plotdatalist(samplelist, labels=['Data', 'Kappa=20', 'Kappa=5'], proj='lambert',
                 mflag = True)
```

figure. Both functions can also calculate and display the median and its confidence cone.

`sphstat` can generate samples from uniform, Fisher, Fisher-Bingham, Kent, and Watson distributions with given parameters. Note that, while Watson distribution is a line distribution, `sphstat` in its present version does not implement tests or estimators for axial distributions.

Descriptive statistics of the sample can be calculated using `resultants` in `sphstat.descriptives`. These include the directional cosines, resultant vector, resultant length, mean direction and mean resultant length, that are returned in a dictionary, which can be displayed using `prettyprintdict` in `sphstat.utils`.

Listing 1 shows the basic operations of reading a sample originally stored in declination/inclination coordinates, generating two samples of 100 observations from a Fisher distribution with different mean directions and concentration parameters, calculating and displaying the resultant statistics of one of these samples, and plotting the three samples in the same figure using Mollweide projection as shown in Fig. 1. Notice that the data points are mapped to $0 \leq \theta \leq \pi$ and $0 \leq \phi < 2\pi$. The data imported in this example pertains to measurements of the direction of magnetisation in specimens from the Great Whin Sill as reported in [28].

5.2. Example 2: Analysis of a single sample

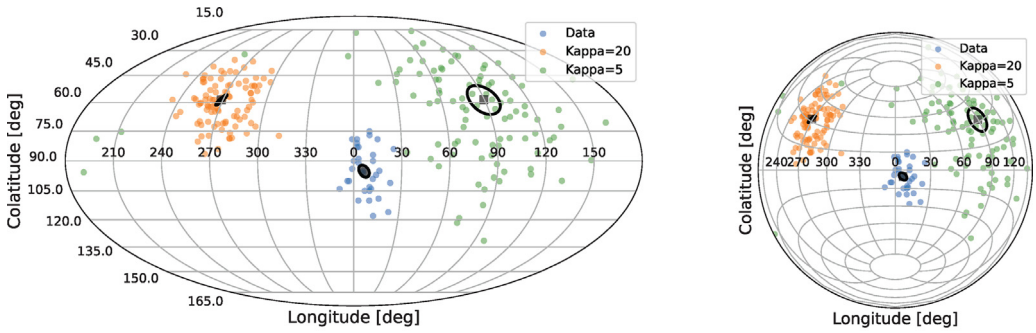
`sphstat` provides functions for the analysis of a single sample of observations. `sphstat.singlesample` comprises several tests and estimators. For example, it is possible to test if the sample comes from a uniform distribution or to test if the population mean is a given prescribed value. The functions in `sphstat.singlesample` include two groups: hypothesis tests and parameter estimators. The hypothesis tests are tests for uniformity (i.e. `isuniform`), rotational symmetry (i.e. `isaxisymmetric`), Fisherianness (i.e. `isfisher`), a test for selecting a Kent distribution as opposed to a Fisher distribution (i.e. `isfishervskent`), testing against a specified mean direction (i.e. `testagainstmean` and `meantest`), median direction (i.e. `testagainstmedian`), or a concentration parameter (i.e. `kappatest`) and an outlier test (i.e. `outliertest`). Single sample parameter estimators include estimators for mean direction (i.e. `meanifsymmetric`), parameter estimators for the Fisher (i.e. `fisherparams`), Kent (i.e. `kentparams`, `kentmeancone`) and Wood (i.e. `bimodalparams`) distributions. The parameter estimators should be used after a specific model is chosen.

Listing 2 shows a pipeline for testing whether is sample can be assumed to be drawn from a spherical uniform distribution or a Fisher distribution. Once the Fisherian hypothesis is retained, the parameters of a Fisherian model for the sample is calculated, and the sample is then tested against a mean direction. The data imported in this example pertains to measurements of magnetic remanence in specimens of Palaeozoic red-beds from Argentina reported in [29].

5.3. Example 3: Analysis of two or more samples

It is often of interest to compare two or more samples for their mean or median directions. The tests `iscommonmedian`, `iscommonmean`, and `isfishercommonmean` are similar to independent samples t-tests and multiple comparisons on more than two linear samples. Similarly, `isfishercommonkappa` is akin to Levene's test [30] or the Brown-Forsythe test [31] for linear samples. The package also includes the functions to calculate the common mean (i.e. `pooledmean` and `fishercommonmean`), and the common concentration parameter (i.e. `fishercommonkappa`) of multiple samples.

In the code fragment given in Listing 3 uses of different functions from this module for analysing a set of four samples are shown. The data imported in this example comprises measurements of occupational judgments according to 4 different criteria (earnings, social



(a) Mollweide projection (b) Lambert projection

Fig. 1. Plots generated using the code from Listing 1 showing the two available projections.

Listing 2
Analysing a single sample using `sphstat.singlesample`.

```
from sphstat.singlesample import *
from sphstat.utils import prettyprintdict, convert
from numpy import deg2rad

sample = readsample('PATH/TO/data.xlsx', typ='rad')
samplerad = convert(sample, 'decinc')
samplecart = polartocart(samplec)
# Check if the sample comes from the spherical uniform distribution
# at alpha=0.01 level
res = isuniform(samplecart, alpha=0.01)
prettyprintdict(res)

teststat:  76.62909931082903
crange:  11.344866730144373
testresult:  False

# Test if the sample is Fisherian at alpha=0.01 level
res = isfisher(samplecart, alpha=0.01)
prettyprintdict(res)

colatitute:  {'stat': 0.5618694195338871, 'crange': 1.308, 'H0': True}
longitude:  {'stat': 0.7798287151581244, 'crange': 1.347, 'H0': True}
twovariable:  {'stat': 0.5456941716956957, 'crange': 1.035, 'H0': True}
H0:  True
alpha:  0.01

# Calculate parameters of Fisher distribution and the 99% CI of kappa estimate
res = fisherparams(samplecart, alpha=0.01)
prettyprintdict(res)

mdir:  (0.572550980653295, 2.516674825714254)
kappa:  113.291584366271
thetaalpha:  0.060029726709833156
cikappa:  (60.98300551288134, 173.18356926438068)

# Check if the population mean is 30 deg. colatitude and 140 deg.  longitude
# at alpha=0.01 level
res = ssmp.testagainstmean(samplecart, [deg2rad(30.), deg2rad(140.)], alpha=0.01)
prettyprintdict(res)

hn:  5.61175442788711
cval:  4.605170185988091
res:  False
```

status, reward, social usefulness), each response cast into a unit norm vector [32].

5.4. Example 4: Correlation, regression, temporal association

`sphstat` also includes modelling tools for calculating the correlation between two samples (i.e. `samplecorrelation`), estimating the population correlation (i.e. `xcorrrandomsamples`, `jackknife_corrci`) calculating the correlation a sample and a circular random variable (i.e. `xcorrsamplevariable`), fitting a regression model for a circular variable (i.e. `regresscircular`), and assessing whether a series of observations are temporally associated (i.e. `isnottemporallyassociated`).

5.4.1. Correlation

In many cases, samples of observations of the same phenomenon on the unit sphere are correlated and the level and direction (i.e. positive, negative or no correlation) of their correlation might be of interest. For

Listing 3
Analysing multiple samples using `sphstat.twosample`.

```
import sphstat.twosample as twsmp
from sphstat.utils import prettyprintdict, readsample, polartocart

# Import the samples stored in different worksheets of a file.
sample1 = readsample('/PATH/TO/data.xlsx', wsindex = 0, typ = 'deg')
sample2 = readsample('/PATH/TO/data.xlsx', wsindex = 1, typ = 'deg')
sample3 = readsample('/PATH/TO/data.xlsx', wsindex = 2, typ = 'deg')
sample4 = readsample('/PATH/TO/data.xlsx', wsindex = 2, typ = 'deg')
slist = [sample1, sample2, sample3]
slistcart = []
for s in list:
    slistcart.append(polartocart(s))
# Check if all samples have a common mean direction
res = twsmp.iscommonmedian(slistcart, True, 0.05)
prettyprintdict(res)

Z2:  262.03788201644767
cval:  12.591587243743977
testresult:  False

# Assuming Fisherian data, do all samples have the same kappa?
res = twsmp.isfishercommonkappa(slistcart, alpha=0.01)
prettyprintdict(res)

Z:  2.532482197515748
cval:  1.959999996736
df:  {'nu': 94, 'r': 4}
testresult:  False

# Assuming Fisherian data, what is the common kappa?
kappahat, klim = twsmp.fishercommonkappa(slistcart, alpha=0.01)
print('kappahat: ', kappahat)
print('95% CI for kappahat: ', klim)

kappahat:  6.429107278709324
95% CI for kappahat:  (5.542904290393356, 7.380065741447061)
```

example, it might be of interest to calculate the correlation between the eye fixations on the same target between two different subjects. The example usage of `sphstat` in analysing correlation is shown in Listing 4. The data imported in this example comprises measurements of magnetic remanence after successive partial demagnetisation stages at different temperatures in specimens of Mesozoic dolerite and is tabulated in [20, Appendix B8, p287].

5.4.2. Regression

Another use case for `sphstat.modelling` is finding a linear regression model for a variable given a set of associated vector observations. `regresscircular` provides a regression model in the form of a function object for predicting points on the unit sphere given a circular random variable. Listing 5 provides a simple example for the usage of `sphstat` for this purpose where a sample from a Kent distribution is generated. The circular variable to be used as the predictor is the angle between the vector and the x-axis. Fig. 2 shows the output of the code in the Listing.

5.4.3. Temporal association

Finally, `sphstat` also provides the functionality to assess whether a time ordered set of observations are temporally associated. Such data can occur, for example, in eye tracking experiments measuring fixation. Listing 6 demonstrates the use of `sphstat` for assessing temporal association using data from [33] which comprises the GPS coordinates of an individual Montagu's harrier during its autumn migration in 2009. As would be expected, the result from `isnotseriallyassociated` indicates that we can reject the null hypothesis that the data is independent in favour of temporal association.

6. Impact

The `sphstat` package has already been used in assessing the results from a subjective localisation experiment that involved a pointing task [19]. While inferential statistics (i.e. hypothesis testing) is quite common in psychoacoustics in general and audio quality evaluation in particular, the author is unaware of any other studies using a statistically grounded approach that `sphstat` affords. This is possibly due to the lack of software tools, open source or otherwise, that implement such functionality. `sphstat` would facilitate how the results of such experiments are analysed and reported. The package can also be used in topics such as room acoustics (e.g. to assess the directional properties of diffuse sound field from energetic measurements), array signal processing (e.g. to analyse direction or arrival estimates), earth sciences (e.g. for the analysis of magnetisation of different deposits), ecology (e.g. for the analysis of biodiversity), astronomy (e.g. for the analysis of data from radiotelescopes), cognitive science and psychology (e.g. for the analysis of eye tracking data).

7. Conclusions

This article presented `sphstat` which is a Python package for applying inferential statistics on vector data on the unit sphere. In its first major version, `sphstat` includes functions for random sample generation from a variety of spherical distributions, functions for descriptive statistics, functions for hypothesis testing on samples of vectorial data on the unit sphere, functions for calculating correlations, functions for fitting a regression model to a circular variable, and hypothesis testing for temporal association of a sample.

While the present version of `sphstat` implements the methods and algorithms given in [20], it is planned that it will be extended

Listing 4
Use of `sphstat.modelling` in correlation of spherical data.

```
import sphstat.modelling as mdl
from sphstat.utils import prettyprintdict, readsample, polartocart
from numpy.random import randn

# Import the samples stored in different worksheets of a file.
sample1 = readsample('/PATH/TO/data.xlsx', wsindex = 0, typ = 'rad')
sample2 = readsample('/PATH/TO/data.xlsx', wsindex = 1, typ = 'rad')
sample1rad = convert(sample1, 'decinc')
sample2rad = convert(sample2, 'decinc')
sample1cart = polartocart(sample1rad)
sample2cart = polartocart(sample2rad)
# Calculate the (population) cross-correlation estimate and test the null
# hypothesis that the samples are uncorrelated
res = xcorrrandomsamples(sampcart1, sampcart2, 10000, htype='=')
prettyprintdict(res)

rhohat:  0.8512942745968715
std:  0.03355675208163771
cval:  (-0.0625971611679344, -0.11443277752692978)
ci:  (0.7855242490787221, 0.9170643001150208)
testresult:  False

# Generate a noisy variable and find the correlation of that variable
# with the sample, also testing the null hypothesis that the sample and
# the variable are uncorrelated
variable = []
axx = np.array([1., 0., 0.])
for ind in range(sampcart1['n']):
    vr = float(np.dot(sampcart1['points'][ind], axx))
    variable.append(randn(4) * 0.1 + vr * np.array([1, 0, 0, 0]))
res = xcorrsamplevariable(sampcart1, variable)
prettyprintdict(res)

rhohatg:  0.23909475670140315
teststat:  44.47162474646099
cval:  21.02606981748307
testresult:  False
```

Listing 5
Regression using `sphstat.modelling`.

```
import sphstat.modelling as mdl
from sphstat.utils import prettyprintdict, carttopolar
from sphstat.distributions import kent
from math import acos
import numpy as np

# Generate a sample of 100 random vectors from the Kent distribution
mu0 = np.array([0., 1., 0.])
samplekentc = kent(100, 50, 20, np.array([1., 1., 1.]), mu0)
samplekentr = carttopolar(samplekentc)
# Generate a random variable which follows the dot product of the Kentian
# sample with the x-axis
vari = []
axx = np.array([1., 0., 0.])
for ind in range(samplekentc['n']):
    vari.append(float(acos(np.dot(samplekentc['points'][ind], axx))))
# Regression model is returned in fitmodel
Uhat, w, alphahat, fitmdl = mdl.regresscircular(samplekentc, vari, alpha=np.pi/2)
# Calculate the regression curve at a number of points
thetas = np.linspace(0, np.pi/2,  180)
sample = dict
sample['points'] = []
sample['type'] = 'cart'
for the in thetas:
    xh = fitmdl(the)
    sample['points'].append(xh)
sample['n'] = 180
sampler = carttopolar(sample)
plotdatalist([samplekentr, sampler], labels=['Data', 'Model'])
```

in future versions to include more recent developments in the field (e.g. [9,34,35]) as well as spherical data in higher dimensions (e.g. [36,37]). Future versions will also include methods and algorithms for axial and girdle distributions and better plotting capabilities.

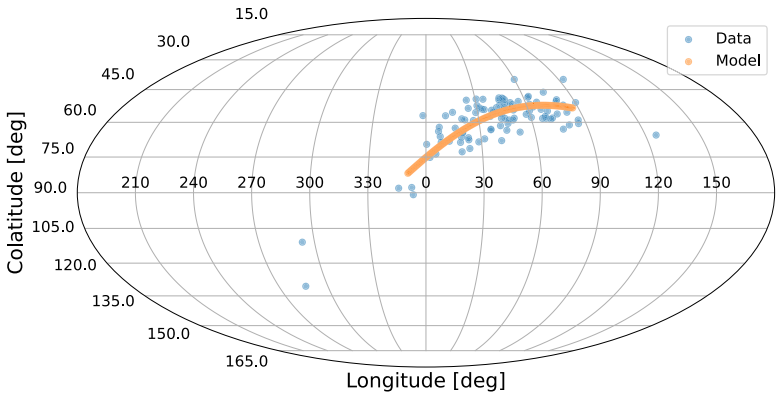


Fig. 2. Plot generated by the code from Listing 5.

Listing 6
Temporal association analysis using `sphstat.modelling`.

```
import sphstat.modelling as mdl
from sphstat.utils import prettyprintdict, polartocart

sample = readsample('/PATH/TO/data.xlsx', typ='rad')
samprad = convert(sample, 'latlon')
sampcart = polartocart(samplerad)
# Test the null hypothesis that the observations in the sample are not
# temporally associated
res = isnotseriallyassociated(sampcart)
prettyprintdict(res)

Sstar:  48.12394425095114
cval:  1.6448536269514722
testresult:  False
```

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Huseyin Hacihabiboglu reports financial support was provided by Scientific and Technological Research Council of Turkey.

Data availability

sphstat Python package is available via PyPI and the code is versioned on GitHub.

Acknowledgements

This work was supported by the Turkish Scientific and Technological Research Council (TUBITAK) Research Grant 119E254 “Audio Signal Processing for Six Degrees of Freedom (6DOF) Immersive Media”.

References

[1] Dumitrache C. On the evolution of filaments. Sol Phys 1997;173(2):281–304.
[2] Populin LC. Human sound localization: measurements in untrained, head-unrestrained subjects using gaze as a pointer. Exp Brain Res 2008;190(1):11–30.
[3] Borradaile G, Henry B. Tectonic applications of magnetic susceptibility and its anisotropy. Earth-Sci Rev 1997;42(1–2):49–93.
[4] Speranza F, Sagnotti L, Mattei M. Tectonics of the Umbria-Marche-Romagna arc (central northern Apennines, Italy): New paleomagnetic constraints. J Geophys Res Solid Earth 1997;102(B2):3153–66.
[5] Ginestet CE, Li J, Balachandran P, Rosenberg S, Kolaczyk ED. Hypothesis testing for network data in functional neuroimaging. Ann Appl Stat 2017;725–50.
[6] Cuartero A, Armesto J, Rodríguez PG, Arias P. Error analysis of terrestrial laser scanning data by means of spherical statistics and 3D graphs. Sensors 2010;10(11):10128–45.
[7] Wenzel EM, Wightman FL, Kistler DJ. Localization with non-individualized virtual acoustic display cues. In: Proceedings of the SIGCHI conference on human factors in computing systems. 1991, p. 351–9.
[8] Schonstein D, Ferré L, Katz BF. Comparison of headphones and equalization for virtual auditory source localization. J Acoust Soc Am 2008;123(5):3724.
[9] Ley C, Verdebout T. Modern directional statistics. Chapman and Hall/CRC; 2017.
[10] Arbuthnot J. An argument for divine providence, taken from the constant regularity observ’d in the births of both sexes. Philos Trans R Soc Lond 1710;27(328):186–90.
[11] Lehmann EL. The Fisher, Neyman-Pearson theories of testing hypotheses: one theory or two? J Am Stat Assoc 1993;88(424):1242–9.
[12] JASP Team. JASP (Version 0.16.4)[Computer software]. 2022, URL https://jasp-stats.org/.
[13] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods 2020;17:261–72. http://dx.doi.org/10.1038/s41592-019-0686-2.
[14] Vallat R. Pingouin: statistics in python. J Open Source Softw 2018;3(31):1026. http://dx.doi.org/10.21105/joss.01026.
[15] Tsagris M, Athineou G, Adam C, Sajib A, Amson E, Waldstein MJ. `Directional`: A collection of functions for directional data analysis. Tech. rep., University of Crete; 2022, URL https://CRAN.R-project.org/package=Directional.
[16] Fryer D, Li M, Olenko A. `rcosmo`: R package for analysis of spherical, HEALPix and cosmological data. R J 2020;12(1).
[17] Fisher RA. Dispersion on a sphere. Proc R Soc A 1953;217(1130):295–305.
[18] Mardia KV. Statistics of directional data. J R Stat Soc Ser B Stat Methodol 1975;37(3):349–71.
[19] Erdem E, Cvetković Z, Hacıhabiboğlu H. 3D perceptual soundfield reconstruction via virtual microphone synthesis. IEEE/ACM Trans Audio Speech Lang Process 2023;31:1305–17.
[20] Fisher NI, Lewis T, Embleton BJ. Statistical analysis of spherical data. Cambridge University Press; 1993.
[21] Van Rossum G, Drake FL. Python 3 reference manual. Scotts Valley, CA: CreateSpace; 2009.
[22] Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature 2020;585:357–62. http://dx.doi.org/10.1038/s41586-020-2649-2.

[23] Meurer A, Smith CP, Paprocki M, Čertík O, Kirpichev SB, Rocklin M, et al. SymPy: symbolic computing in Python. PeerJ Comput Sci 2017;3:e103.
[24] McKinney W, et al. Data structures for statistical computing in python. In: Proceedings of the 9th Python in science conference, vol. 445. Austin, TX; 2010, p. 51–6.
[25] Gazoni E, Clark C. openpyxl: A Python library to read/write Excel 2010 xlsx/xlsm files. 2018, http://openpyxl.readthedocs.org/en/default.
[26] Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng 2007;9(3):90–5.
[27] Brandl G. Sphinx documentation. 2021, http://sphinx-doc.org/sphinx.pdf.
[28] Creer K, Irving E, Nairn A. Palaeomagnetism of the Great Whin sill. Geophys J Int 1959;2(4):306–23.
[29] Embleton B. Palaeomagnetic results for the Permian of South America and a comparison with the African and Australian data. Geophys J Int 1970;21(2):105–18.
[30] Levene H. Robust tests for equality of variances. In: Contributions to probability and statistics. Essays in honor of Harold Hotelling. Stanford University Press; 1961, p. 279–92.
[31] Brown MB, Forsythe AB. Robust tests for the equality of variances. J Amer Statist Assoc 1974;69(346):364–7.
[32] Holguâin J. The application of directional methods in p dimensions. [Master's thesis], Canada: Simon Fraser University, Dept. of Mathematics; 1980.
[33] Schlaich AE, Bouten W, Bretagnolle V, Heldbjerg H, Klaassen RH, Sørensen IH, et al. A circannual perspective on daily and total flight distances in a long-distance migratory raptor, the Montagu's harrier, Circus pygargus. Biol Lett 2017;13(6):20170073.
[34] Ley C, Verdebout T. Applied directional statistics: Modern methods and case studies. CRC Press; 2018.
[35] Pewsey A, García-Portugués E. Recent advances in directional statistics. Test 2021;30(1):1–58.
[36] Harman R, Lacko V. On decompositional algorithms for uniform sampling from n-spheres and n-balls. J Multivariate Anal 2010;101(10):2297–304.
[37] You K, Suh C. Parameter estimation and model-based clustering with spherical normal distribution on the unit hypersphere. Comput Statist Data Anal 2022;171:107457.