circular: an R package for circular data analysis

Claudio Agostinelli
claudio@unive.it

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca' Foscari di Venezia
http://www.dst.unive.it/~claudio

22 May 2014

Outline

`circular` and `CircStats` packages I

`CircStats`

- A porting of original codes for S-plus by Ulric Lund to R.
- Used in the book by S. Rao Jammalamadaka and A. SenGupta (2001) Topics in circular Statistics, World Scientific.
- Several improvements and bug fixed.
- Actual version is 0.2-4 (2012-10-29)
- On CRAN since 2002-02-21, but still there only for historical reasons.
- Updated only for bug fix and for keep it compatible with new versions of R .

`circular` and `CircStats` packages II

`circular`

- The package is implemented in the S3 object system.
- Many new function available since `CircStats`.
- Actual version (on CRAN) is 0.4-7 (2013-11-08)
- On CRAN since 2004/05/26.
- On R-Forge since 2007/11/06 at r-forge.r-project.org/projects/circular, please join us.
- Well documented in the book by A. Pewsey, M. Neuhäuser, and G.D. Ruxton (2013) Circular Statistics in R , Oxford University Press.

circular

It produces an object of class `circular` with attribute `circularp`

```
x <- circular(matrix(c(0, 90, 180, 270),
    nrow = 2), units = "degrees")
class(x)
```

```
[1] "circular" "matrix"
```

```
attr(x, "circularp")
```

```
$type = "angles"
$units = "degrees"
$template = "none"
$modulo = "asis"
$zero = 0
$rotation = "counter"
```

conversion.circular

It allows to convert from one coordinate/units system to another one.

```
print(y <- conversion.circular(x, units = "radians",
    rotation = "clock"))
```

```
Circular Data:
Type = angles
Units = radians
Template = none
Modulo = asis
Zero = 0
Rotation = clock
       [,1]   [,2]
[1,]  0.000 -3.142
[2,] -1.571 -4.712
```

It is used internally to each function for performing conversions.

Other important functions

- `mean`, `var`, `sd`, `summary`, `median`, `quantile`.
- `[rdpq]vonmises`, `[rdpq]wrappednormal`, `[rdpq]cauchy`, and many more.
- `plot`, `points`, `lines`, `rose.diag`, `windrose`, `heatmap`.

Few words about the structure of the functions

- Each function (e.g. `mean.circular`) is a wrapper for a workhorse function which is hidden to the final user (e.g. `MeanCircularRad`).
- The wrapper (`mean.circular`):
  - handle missing values;
  - convert the data in the appropriate unit/coordinate system
  - if necessary reconvert the output in the appropriate unit/coordinate system
- The workhorse (`MeanCircularRad`):
  - performs the calculation (or the plot, etc.) in a given unit/coordinate system
  - by convention the name is CamelCase and the last three digits are Rad or Deg.
  - they are used inside other workhorses or wrappers. The wrapped are not called inside the workhorses.

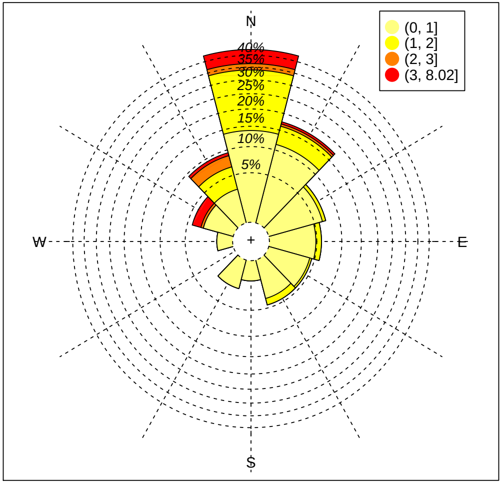
Wind data set

```r
library(circular)
dataset <- read.table(file = "./wind.dat",
    sep = ";", header = TRUE)
mag <- dataset$mag
dir <- dataset$dir
ok <- complete.cases(dir, mag)
mag <- mag[ok]
magt <- mag
magt[magt > 3] <- 3.05
dir <- circular(dir[ok], units = "degrees",
    template = "geographics")
```

Summary statistics

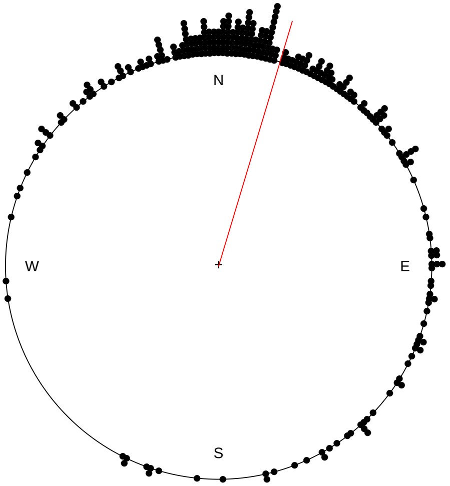
```
mean(dir)

Circular Data:
Type = angles
Units = degrees
Template = geographics
Modulo = asis
Zero = 1.571
Rotation = clock
[1] 16.74

sd(dir)

[1] 0.9187

var(dir)

[1] 0.3443
```

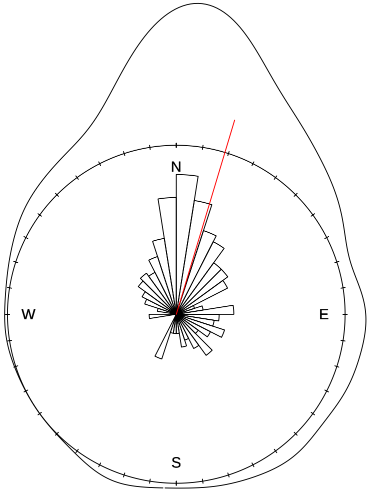
```
median(dir)

Circular Data:
Type = angles
Units = degrees
Template = geographics
Modulo = asis
Zero = 1.571
Rotation = clock
[1] 9.48
attr(,"medians")
[1] 9.33 9.63

quantile(dir, c(0.25, 0.5, 0.75))

Circular Data:
Type = angles
Units = degrees
```



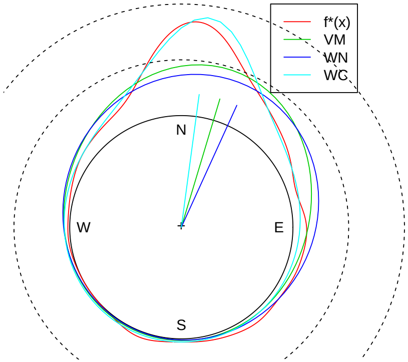
Wind rose of the dataset from Col de la Roa (Italy) meteorological station in March and April 2001 between 3.00 and 4.00 o'clock (n=310).

▸ R code





Stack Circular Plot (Left) and estimated density (Right, together with a Rose Diagram) of the Wind direction dataset. ▸ R code



MLE estimators for the Wind direction dataset using

- Von Mises (VM, $\hat{\mu} = 16.74$, $\hat{\kappa} = 1.76$);
- Wrapped Normal (WN, $\hat{\mu} = 24.487$, $\hat{\rho} = 0.603$);
- Wrapped Cauchy (WC, $\hat{\mu} = 7.662$, $\hat{\rho} = 0.697$). R code.

Robust Statistics

Robust Statistics is a discipline which deals with the inference problem in the case of misspecification of the model with respect to the distribution of the data. A “robust” estimator is able to well describe the parameters, or some features, of the “majority” (or “bulk”) of the data. And to quote:

- Hampel et al. [1986], pag. 56: “whereas in classical statistics the model has to fit all the data, in robust statistics it *may* be enough that it fit the majority of the data, the remainder being regarded as outliers”;

Some references: Collett [1980], Lenth [1981], Upton [1993] and SenGupta and Laha [2001]. A survey is He [1992].

Outliers in Circular data

- In Euclidean setting the outliers are defined as:
  - Barnett and Lewis [1994], pag. 7: "... an outliers in a set of data to be an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data";
  - Hawkins [1980]: an outlier is "an observation that deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism".

  The definition is based on some **"geometric" distance** between observations.
- In Circular setting: the sample space is bounded and the parametric space is often bounded too (e.g. the parametric space of the mean direction in the Von Mises distribution is every interval of length 2π, in general, $[0, 2\pi)$).

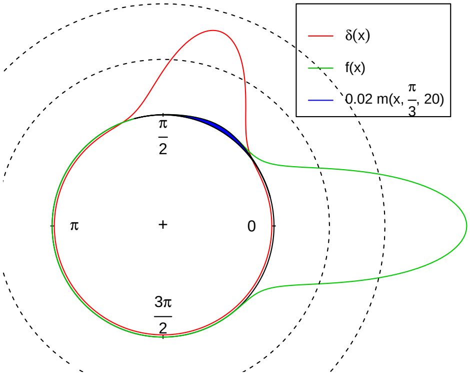
Outliers in Circular Data: Pearson Residuals

In our approach, outliers are observations that are highly unlikely to occur under the assumed model [see Markatou et al., 1995].

This definition is well adapted in the circle since it is based on a **“probabilistic” distance**. One way to measure this discrepancy is to use the Pearson Residuals [Lindsay, 1994] defined as follows

$$\delta(x, \theta, f^*) = \frac{f^*(x)}{m^*(x;\theta)} - 1$$

where $f^*(x)$ is a non parametric density estimator based on the data and $m^*(x;\theta)$ is a smoothed version of the density of the model. Note that $\delta(x, \theta, f^*) = \delta(x \mod (2\pi), \theta, f^*)$.



Pearson Residuals ($\delta(x; \theta = (0, 5), f(x))$) for $f(x) = 0.98m(x; 0, 5) + 0.02m(x; \pi/3, 20)$ where $m(x; \mu, \kappa)$ is the density of a Von Mises distribution. ▸ R code

Minimum Distance Estimators

For continuous models the Power Divergence Measure [Cressie and Read, 1988] between the densities $f^*(x)$, $m(x,\theta)$ is

$$\int_0^{2\pi} \frac{\left(\frac{f^*(x)}{m(x;\theta)}\right)^{\alpha+1}}{\alpha\ (\alpha+1)}\ m(x;\theta)\ \mathrm{d}x$$

or using the Pearson Residual function (in the unsmoothed model) $\delta(x;\theta,f^*)$:

$$\int_0^{2\pi} G(\delta(x;\theta,f^*))m(x;\theta)\ \mathrm{d}x$$

where $G(\delta(x;\theta,f^*)) = \frac{(\delta(x;\theta,f^*)+1)^{\alpha+1}}{\alpha\ (\alpha+1)}$.
Examples are:

- $\alpha = -1/2$: Hellinger distance;
- $\alpha \to -1$: Kullback–Leibler divergence;
- $\alpha = -2$: Neyman's Chi–Square.

Numerical Calculation of Distances

Since $f^*(x)$ and $m^*(x;\theta)$ and their transformations are 2π periodic functions the Power Divergence Measure is computed easily and fast by Fast Fourier Transform and Parseval's formula. In fact, let $\phi_k(\cdot)$ the Fourier transform, we get

$$2\pi \int_0^{2\pi} f^*(x)^{\alpha+1}\, m^*(x;\theta)^{-\alpha}\,\mathrm{d}x = \sum_{k=-\infty}^{\infty} \phi_k\left(f^*(x)^{\alpha+1}\right)\phi_k\left(m^*(x;\theta)^{-\alpha}\right)$$

Weighted Likelihood Estimating Equations

The estimating equations of WLEE is a modified version of the MLE equations where at each score is associated a weight defined as follows

$$w(x;\theta,f^*) = \frac{A(\delta(x;\theta,f^*)) + 1}{\delta(x;\theta,f^*) + 1}$$

where $A(\delta)$ is the Residual Adjustment Function, with the form related to the Power Divergence Measure given by

$$A(\delta) = \frac{(\delta + 1)^{\alpha+1} - 1}{\alpha + 1}$$

Hence the WLEE estimator is the solution of

$$\sum_{i=1}^{n} w(x_i;\theta,f^*)u(x_i;\theta) = 0$$

where $u(x;\theta)$ is the score function for the model.

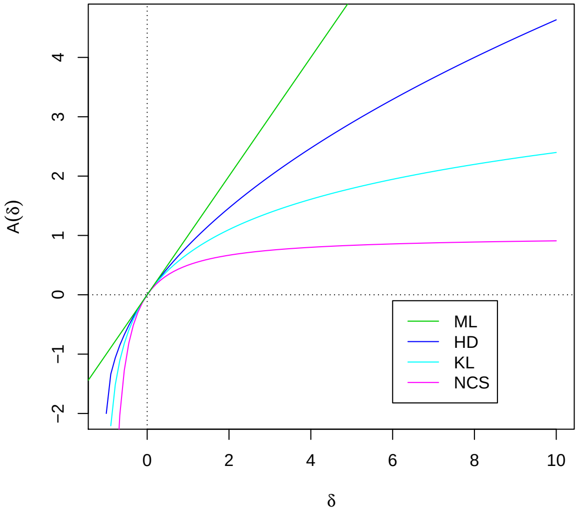
Example: Von Mises distribution

The WLEE for the Von Mises distribution is

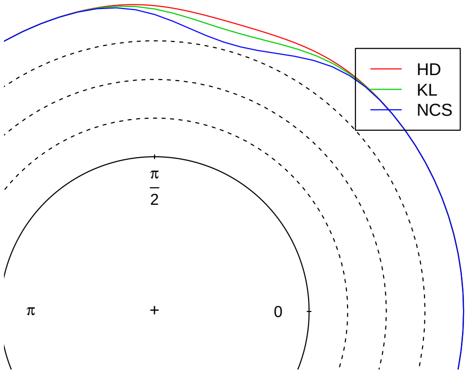
$$\begin{cases} \mu = \arctan^* \left(\frac{\sum_{i=1}^n w(x_i;\mu,\kappa) \ \sin(x_i)}{\sum_{i=1}^n w(x_i;\mu,\kappa) \ \cos(x_i)} \right) \\ \kappa = \mathrm{A}^{-1} \left(\frac{\sum_{i=1}^n w(x_i;\mu,\kappa) \ \cos(x_i-\mu)}{\sum_{i=1}^n w(x_i;\mu,\kappa)} \right) \end{cases}$$

where

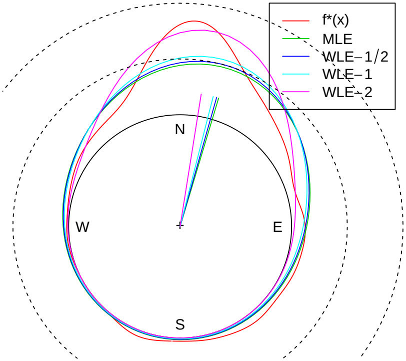
- $\mathrm{A}(\kappa) = \mathrm{I}_1(\kappa)/\mathrm{I}_0(\kappa)$ is the ratio of the two modified Bessel functions of the first kind with order zero and one;
- $\arctan^*$ is the “quadrant-specific” inverse of the tangent that provides the unique inverse of the tangent in $[0, 2\pi)$.



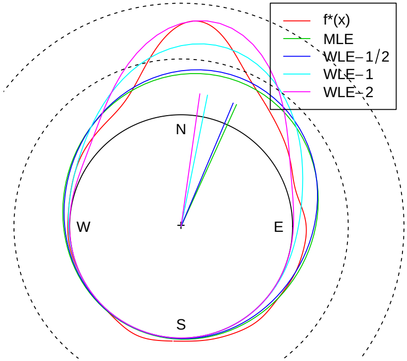
Residual Adjustment Function: Hellinger (HD), Kullback–Leibler (KL), Neyman's Chi–square (NCS) ▸ R code



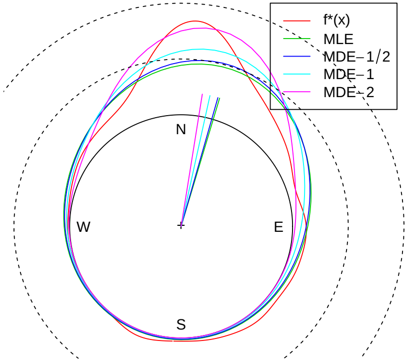
Weight function $(w(x; \theta = (0, 5), f(x)))$ for $f(x) = 0.98m(x; 0, 5) + 0.02m(x; \pi/3, 20)$ where $m(x; \mu, \kappa)$ is the density of a Von Mises distribution. For Residual Adjustment Function: Hellinger (HD), Kullback–Leibler (KL), Neyman's Chi–square (NCS) ▸ R code



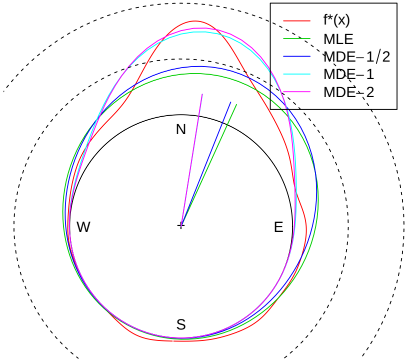
WLE estimators for the Wind direction dataset using the Von Mises model. ▸ R code



WLE estimators for the Wind direction dataset using the Wrapped Normal model. ▸ R code



MDE estimators for the Wind direction dataset using the Von Mises model. ▸ R code



MDE estimators for the Wind direction dataset using the Wrapped Normal model. R code

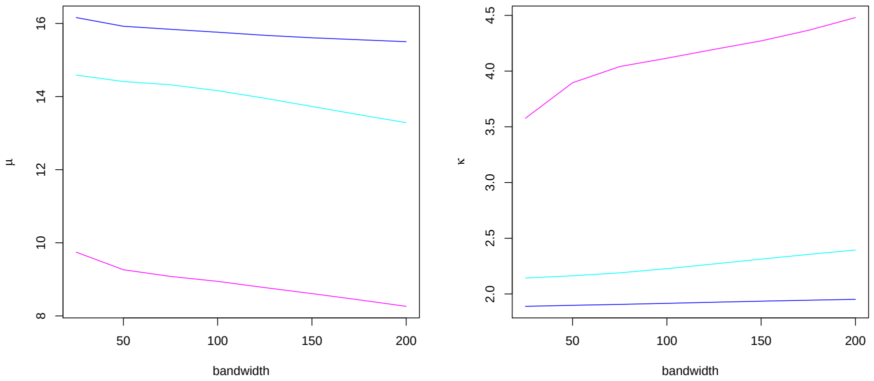
Results for the Wind dataset

Von Mises Model

	MLE	MDE -1/2	MDE -1	MDE -2	WLE -1/2	WLE -1	WLE -2
μ	16.740	15.966	12.456	9.124	15.840	14.322	9.081
κ	1.760	1.961	2.631	4.228	1.907	2.190	4.041

Wrapped Normal Model

	MLE	MDE -1/2	MDE -1	MDE -2	WLE -1/2	WLE -1	WLE -2
μ	24.487	21.787	9.156	9.084	22.920	11.381	8.055
σ	1.005	0.859	0.526	0.504	0.922	0.611	0.466



WLE estimators for the Wind direction dataset using the Von Mises model and different bandwidth. ▸ R code

Distances between circular data sets

The function `dist.circular` similar to the function `dist` provides distances for a `data.frame` or a `matrix` of circular data. The following distances are available

- `correlation` $d(\mathbf{x}, \mathbf{y}) = \sqrt{1 - \rho(\mathbf{x}, \mathbf{y})}$ where

$$\rho(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n}(sin(x_i - \mu_{\mathbf{x}})sin(y_i - \mu_{\mathbf{y}}))}{(\sum_{i=1}^{n}(sin(x_i - \mu_{\mathbf{x}})^2)\sum_{i=1}^{n}(sin(y_i - \mu_{\mathbf{y}})^2))^{1/2}}$$

  is the correlation coefficient between the two vectors $\mathbf{x}$ and $\mathbf{y}$.

- `angularseparation`

$$d(\mathbf{x}, \mathbf{y}) = sum_{i=1}^{n}(1 - cos(x_i - y_i))$$

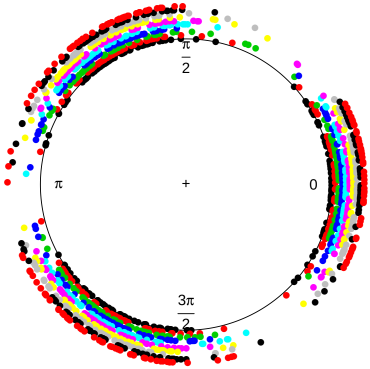
- `chord`

$$d(\mathbf{x}, \mathbf{y}) = sum_{i=1}^{n}\sqrt{2(1 - cos(x_i - y_i))}$$

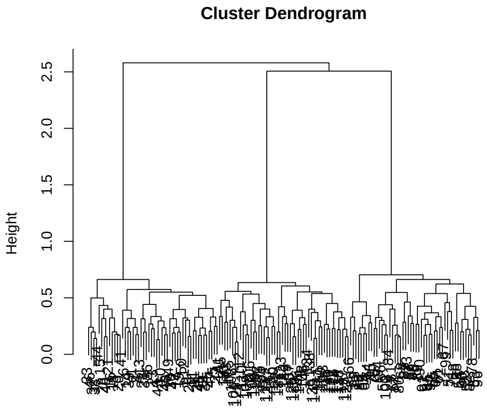
- `geodesic`

$$\sum_{i=1}^{n} \pi - |\pi - |x_i - y_i||$$

  where the $|\cdot|$ is expressed with an angle in $[-\pi, \pi]$.

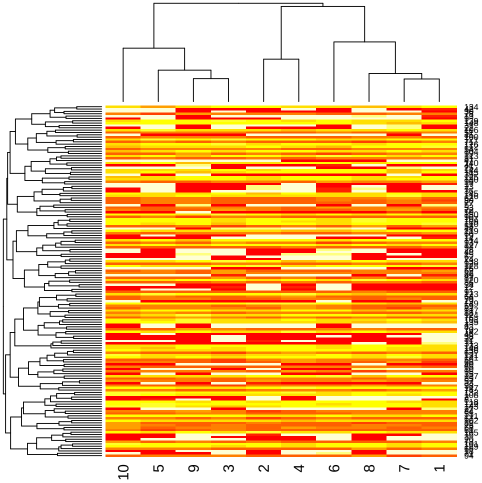


A random circular data set with 50 observations and 10 variables.



dist.circular(X, method = "geodesic")
hclust (*, "complete")

Dendrogram given by cluster analysis using `geodesic` distance. ▸ R code



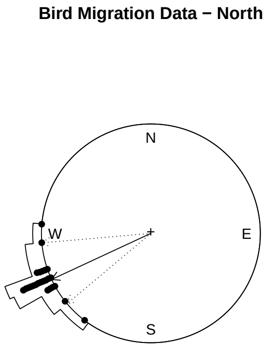
Heatmap. ▸ R code

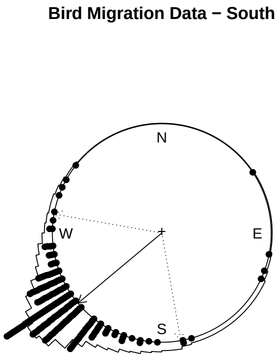
Migration Birds

- Directions of bird migrations can be derived from comparison of bird recovery site with ringing site, during migrations.
- Here the data illustrated in Fransson [2001] are considered.
- They refer to specimens of *Accipiter nisus* found during the first autumn migration and ringed as nestlings in different parts of Sweden.
- Sample units are divided into two groups, according to whether ringing site is northern or southern Sweden. They are referred to as North and South, respectively.

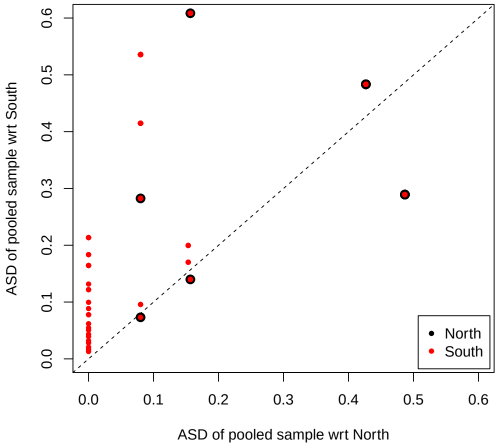
Statistic	North	South
Sample size	25	165
Mean, $\bar{x}_n$	4.24	4.09
Mean resultant length, $\bar{R}_n$	0.984	0.897
Angular median, $\widehat{\alpha}_{0.5}$	4.23	4.14
Mean deviation, $\widehat{\delta}_n$	0.0977	0.307
ASD dispersion statistic, $\widehat{\gamma}_n^{(S)}$	0.172	0.480
50% ASD region, $\widehat{D}_n(0.5)$	$(4.14, 4.32)$	$(3.89, 4.33)$

Accipiter nisus migration. Summary statistics (radians).


Bird Migration Data − North
N
W
E
S


Bird Migration Data − South
N
W
E
S

R code



D-D Plot Bird Migration North & South ▸ R code

Conclusions

- We introduced main functions available in the package `circular`
- We introduced Robust estimators for circular data based on
  - Minimum Distance
  - Weighted Likelihood
- We introduced (local) Data Depth for circular data based on
  - (local) Angular simplicial depth
  - (local) Angular halfspace depth
- We used, mainly, packages `circular`, `localdepth` and `wle`.

```r
library(circular)
dataset <- read.table(file = "./wind.dat",
    sep = ";", header = TRUE)
mag <- dataset$mag
dir <- dataset$dir
ok <- complete.cases(dir, mag)
mag <- mag[ok]
magt <- mag
magt[magt > 3] <- 3.05
dir <- circular(dir[ok], units = "degrees",
    template = "geographics")
```

```r
breaks <- circular(0:11/6 * pi - pi/12)

windrose(x = dir, y = magt, increment = 1,
    breaks = breaks, fill.col = rev(heat.colors(4)),
    label.freq = TRUE, main = "")
legend(x = 0.7, y = 1.2, legend = c("(0, 1]",
    "(1, 2]", "(2, 3]", "(3, 8.02]"), pch = 19,
    col = rev(heat.colors(4)), pt.cex = 2)
```

◂ Return

```r
plot(dir, stack = TRUE, ylim = c(-1, 1.22),
    frame = TRUE)
mc <- mean(dir)
lines(c(mc, mc), c(-1, 0.2), col = 2)
plot(density(dir, bw = 40), ticks = FALSE,
    ylim = c(-1, 1.8), main = "", xlab = "",
    ylab = "")
rose.diag(dir, prop = 2, bins = 40, add = TRUE)
lines(c(mc, mc), c(-1, 0.2), col = 2)
```

◂ Return

```r
mvm <- mle.vonmises(dir)
mwn <- mle.wrappednormal(dir)
mwc <- mle.wrappedcauchy(dir)

plot(density(dir, bw = 40), ticks = FALSE,
    xlim = c(-0.8, 1.4), ylim = c(-1, 1.94),
    main = "", xlab = "", ylab = "", col = 2)
plot.function.circular(function(x) dvonmises(x,
    mu = mvm$mu, kappa = mvm$kappa), join = TRUE,
    add = TRUE, col = 3)
plot.function.circular(function(x) dwrappednormal(x,
    mu = mwn$mu, rho = mwn$rho), join = TRUE,
    add = TRUE, col = 4)
plot.function.circular(function(x) dwrappedcauchy(x,
    mu = mwc$mu, rho = mwc$rho), join = TRUE,
    add = TRUE, col = 5)
lines(c(mvm$mu, mvm$mu), c(-1, 0.2), col = 3)
lines(c(mwn$mu, mwn$mu), c(-1, 0.2), col = 4)
lines(c(mwc$mu, mwc$mu), c(-1, 0.2), col = 5)
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(1, 100), lty = 2, col = 1)
legend(0.8, 2, legend = c("f*(x)", "VM",
    "WN", "WC"), col = 2:5, lty = rep(1,
    4))
```

◂ Return

```
delta <- function(x, eps = 0.02) {
    res <- ((1 - eps) * dvonmises(x, circular(0),
        5) + eps * dvonmises(x, circular(pi/3),
        20))/dvonmises(x, circular(0), 5) -
        1
    return(res)
}

curve.circular(delta, n = 501, join = TRUE,
    xlim = c(-1.2, 2.7), ylim = c(-0.6, 1.8),
    col = 2, tcl.text = 0.2, cex = 0.9)
plot.function.circular(function(x) 0.98 *
    dvonmises(x, circular(0), 20) + 0.02 *
    dvonmises(x, circular(pi/3), 20), n = 501,
    join = TRUE, add = TRUE, col = 3)
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
```

```
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(1, 100), lty = 2, col = 1)
legend(1.2, 2, legend = c(expression(delta(x)),
    "f(x)", expression(paste("0.02 m(x, ",
        frac(pi, 3), ", 20)", sep = ""))),
    lty = rep(1, 3), col = c(2, 3, 4), cex = 0.8)

cont <- function(x) {
    den <- 0.02 * dvonmises(x, circular(pi/3),
        20)
    y <- c((1 + den) * sin(x), rev(sin(x)))
    x <- c((1 + den) * cos(x), rev(cos(x)))
    attr(x, "circularp") <- attr(y, "circularp") <- NULL
    res <- list(x = x, y = y)
    return(res)
}

res <- cont(circular(seq(pi/3 - pi/4, pi/3 +
    pi/4, 0.01)))
polygon(x = res, col = 4)
```

◂ Return

```
A <- function(x, alpha) {
    if (alpha == -1)
        a <- log(x + 1) else a <- ((x + 1)^(alpha + 1) - 1
        1)
    return(a)
}

plot(function(x) A(x, alpha = -1/2), from = -1,
    to = 10, xlab = expression(delta), ylab = expression(A
    col = 4)
plot(function(x) A(x, alpha = -1), from = -1,
    to = 10, col = 5, add = TRUE)
plot(function(x) A(x, alpha = -2), from = -1,
    to = 10, col = 6, add = TRUE)
abline(0, 1, col = 3)
abline(h = 0, lty = 3)
abline(v = 0, lty = 3)
```

```
legend(6, -0.1, legend = c("ML", "HD", "KL",
    "NCS"), col = 3:6, lty = rep(1, 4))
```

◂ Return

```
w <- function(x, alpha) {
    w <- (A(x, alpha) + 1)/(x + 1)
    w[w > 1] <- 1
    w[w < 0] <- 0
    return(w)
}

weight <- function(x, alpha = -1/2) {
    w(delta(x), alpha)
}

curve.circular(weight, n = 501, join = TRUE,
    xlim = c(-0.8, 2), ylim = c(0, 2), col = 2,
    tcl.text = 0.2, cex = 0.9)
plot.function.circular(function(x) weight(x,
    alpha = -1), add = TRUE, col = 3)
plot.function.circular(function(x) weight(x,
    alpha = -2), add = TRUE, col = 4)
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.25, 100), lty = 2, col = 1)
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.75, 100), lty = 2, col = 1)
legend(1.3, 1.7, legend = c("HD", "KL", "NCS"),
    col = 2:4, lty = rep(1, 3))
```

◂ Return

```
wvmhd <- wle.vonmises(dir, smooth = 75, group = 5,
    alpha = -1/2)
wvmkl <- wle.vonmises(dir, smooth = 75, group = 5,
    alpha = -1)
wvmncs <- wle.vonmises(dir, smooth = 75,
    group = 5, alpha = -2)
plot(density(dir, bw = 40), ticks = FALSE,
    xlim = c(-0.8, 1.4), ylim = c(-1, 1.94),
    main = "", xlab = "", ylab = "", col = 2)

plot.function.circular(function(x) dvonmises(x,
    mu = mvm$mu, kappa = mvm$kappa), join = TRUE,
    add = TRUE, col = 3)
lines(c(mvm$mu, mvm$mu), c(-1, 0.2), col = 3)

plot.function.circular(function(x) dvonmises(x,
    mu = wvmhd$mu, kappa = wvmhd$kappa),
    join = TRUE, add = TRUE, col = 4)
lines(c(wvmhd$mu, wvmhd$mu), c(-1, 0.2),
    col = 4)

plot.function.circular(function(x) dvonmises(x,
    mu = wvmkl$mu, kappa = wvmkl$kappa),
    join = TRUE, add = TRUE, col = 5)
lines(c(wvmkl$mu, wvmkl$mu), c(-1, 0.2),
    col = 5)

plot.function.circular(function(x) dvonmises(x,
    mu = wvmncs$mu, kappa = wvmncs$kappa),
    join = TRUE, add = TRUE, col = 6)
lines(c(wvmncs$mu, wvmncs$mu), c(-1, 0.2),
    col = 6)

lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
```

```r
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(1, 100), lty = 2, col = 1)
legend(0.8, 2, legend = c("f*(x)", "MLE",
    expression(paste("WLE", -1/2, sep = "")),
    expression(paste("WLE", -1, sep = "")),
    expression(paste("WLE", -2, sep = ""))),
    col = 2:6, lty = rep(1, 5))
```

◂ Return

```
wwnhd <- wle.wrappednormal(dir, smooth = 0.006,
    group = 5, alpha = -1/2)
wwnkl <- wle.wrappednormal(dir, smooth = 0.006,
    group = 5, alpha = -1)
wwnncs <- wle.wrappednormal(dir, smooth = 0.006,
    group = 5, alpha = -2)
plot(density(dir, bw = 40), ticks = FALSE,
    xlim = c(-0.8, 1.4), ylim = c(-1, 1.94),
    main = "", xlab = "", ylab = "", col = 2)

plot.function.circular(function(x) dwrappednormal(x,
    mu = mwn$mu, rho = mwn$rho), join = TRUE,
    add = TRUE, col = 3)
lines(c(mwn$mu, mwn$mu), c(-1, 0.2), col = 3)

plot.function.circular(function(x) dwrappednormal(x,
    mu = wwnhd$mu, rho = wwnhd$rho), join = TRUE,
    add = TRUE, col = 4)
lines(c(wwnhd$mu, wwnhd$mu), c(-1, 0.2),
    col = 4)

plot.function.circular(function(x) dwrappednormal(x,
    mu = wwnkl$mu, rho = wwnkl$rho), join = TRUE,
    add = TRUE, col = 5)
lines(c(wwnkl$mu, wwnkl$mu), c(-1, 0.2),
    col = 5)

plot.function.circular(function(x) dwrappednormal(x,
    mu = wwnncs$mu, rho = wwnncs$rho), join = TRUE,
    add = TRUE, col = 6)
lines(c(wwnncs$mu, wwnncs$mu), c(-1, 0.2),
    col = 6)

lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
```

```r
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(1, 100), lty = 2, col = 1)
legend(0.8, 2, legend = c("f*(x)", "MLE",
    expression(paste("WLE", -1/2, sep = "")),
    expression(paste("WLE", -1, sep = "")),
    expression(paste("WLE", -2, sep = ""))),
    col = 2:6, lty = rep(1, 5))
```

◂ Return

```r
dvmhd <- mde.vonmises(dir, bw = 75, n = 1024,
    alpha = -1/2)
dvmkl <- mde.vonmises(dir, bw = 75, n = 1024,
    alpha = -1)
dvmncs <- mde.vonmises(dir, bw = 75, n = 1024,
    alpha = -2)
plot(density(dir, bw = 40), ticks = FALSE,
    xlim = c(-0.8, 1.4), ylim = c(-1, 1.94),
    main = "", xlab = "", ylab = "", col = 2)

plot.function.circular(function(x) dvonmises(x,
    mu = mvm$mu, kappa = mvm$kappa), join = TRUE,
    add = TRUE, col = 3)
lines(c(mvm$mu, mvm$mu), c(-1, 0.2), col = 3)

plot.function.circular(function(x) dvonmises(x,
    mu = dvmhd$mu, kappa = dvmhd$kappa),
    join = TRUE, add = TRUE, col = 4)
lines(c(dvmhd$mu, dvmhd$mu), c(-1, 0.2),
    col = 4)

plot.function.circular(function(x) dvonmises(x,
    mu = dvmkl$mu, kappa = dvmkl$kappa),
    join = TRUE, add = TRUE, col = 5)
lines(c(dvmkl$mu, dvmkl$mu), c(-1, 0.2),
    col = 5)

plot.function.circular(function(x) dvonmises(x,
    mu = dvmncs$mu, kappa = dvmncs$kappa),
    join = TRUE, add = TRUE, col = 6)
lines(c(dvmncs$mu, dvmncs$mu), c(-1, 0.2),
    col = 6)

lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
```

```r
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(1, 100), lty = 2, col = 1)
legend(0.8, 2, legend = c("f*(x)", "MLE",
    expression(paste("MDE", -1/2, sep = "")),
    expression(paste("MDE", -1, sep = "")),
    expression(paste("MDE", -2, sep = ""))),
    col = 2:6, lty = rep(1, 5))
```

Return

```r
dwnhd <- mde.wrappednormal(dir, bw = 0.0375,
    n = 1024, alpha = -1/2)
dwnkl <- mde.wrappednormal(dir, bw = 0.0375,
    n = 1024, alpha = -1)
dwnncs <- mde.wrappednormal(dir, bw = 0.09,
    n = 1024, alpha = -2)
plot(density(dir, bw = 40), ticks = FALSE,
    xlim = c(-0.8, 1.4), ylim = c(-1, 1.94),
    main = "", xlab = "", ylab = "", col = 2)

plot.function.circular(function(x) dwrappednormal(x,
    mu = mwn$mu, rho = mwn$rho), join = TRUE,
    add = TRUE, col = 3)
lines(c(mwn$mu, mwn$mu), c(-1, 0.2), col = 3)

plot.function.circular(function(x) dwrappednormal(x,
    mu = dwnhd$mu, rho = dwnhd$rho), join = TRUE,
    add = TRUE, col = 4)
lines(c(dwnhd$mu, dwnhd$mu), c(-1, 0.2),
    col = 4)

plot.function.circular(function(x) dwrappednormal(x,
    mu = dwnkl$mu, rho = dwnkl$rho), join = TRUE,
    add = TRUE, col = 5)
lines(c(dwnkl$mu, dwnkl$mu), c(-1, 0.2),
    col = 5)

plot.function.circular(function(x) dwrappednormal(x,
    mu = dwnncs$mu, rho = dwnncs$rho), join = TRUE,
    add = TRUE, col = 6)
lines(c(dwnncs$mu, dwnncs$mu), c(-1, 0.2),
    col = 6)

lines(circular(seq(0, 2 * pi, length = 100)),
    rep(0.5, 100), lty = 2, col = 1)
```

```r
lines(circular(seq(0, 2 * pi, length = 100)),
    rep(1, 100), lty = 2, col = 1)
legend(0.8, 2, legend = c("f*(x)", "MLE",
    expression(paste("MDE", -1/2, sep = "")),
    expression(paste("MDE", -1, sep = "")),
    expression(paste("MDE", -2, sep = ""))),
    col = 2:6, lty = rep(1, 5))
```

◂ Return

```
bw <- seq(25, 200, 25)
temp <- function(x, dir, alpha) wle.vonmises(x = dir,
    group = 5, smooth = x, alpha = alpha)$mu
muhdbw <- sapply(bw, temp, dir = dir, alpha = -1/2)
muklbw <- sapply(bw, temp, dir = dir, alpha = -1)
muncsbw <- sapply(bw, temp, dir = dir, alpha = -2)
plot(bw, muhdbw, xlab = "bandwidth", ylab = expression(mu),
    main = "", type = "l", col = 4, ylim = range(muhdbw,
        muklbw, muncsbw))
lines(bw, muklbw, col = 5)
lines(bw, muncsbw, col = 6)
bw <- seq(25, 200, 25)
temp <- function(x, dir, alpha) wle.vonmises(x = dir,
    group = 5, smooth = x, alpha = alpha)$kappa
kappahdbw <- sapply(bw, temp, dir = dir,
    alpha = -1/2)
kappaklbw <- sapply(bw, temp, dir = dir,
    alpha = -1)
kappancsbw <- sapply(bw, temp, dir = dir,
    alpha = -2)
plot(bw, kappahdbw, xlab = "bandwidth", ylab = expression(k
    main = "", type = "l", col = 4, ylim = range(kappahdbw,
        kappaklbw, kappancsbw))
lines(bw, kappaklbw, col = 5)
lines(bw, kappancsbw, col = 6)
```

◂ Return

```
X <- circular(rbind(matrix(rvonmises(500,
    circular(0), 10), ncol = 10), matrix(rvonmises(500,
    circular(2 * pi/3), 10), ncol = 10),
    matrix(rvonmises(500, circular(-2 * pi/3),
        10), ncol = 10)))
plot(X, xlim = c(-1.125, 1.125), ylim = c(-1.125,
    1.125))
```

◂ Return

```r
h <- hclust(dist.circular(X, method = "geodesic"))
plot(h)
```

◂ Return

```
heatmap.circular(X)
```

◂ Return

```
north <- circular(rep(c(215, 230, 240, 245,
    250, 265, 275), c(1, 1, 4, 12, 5, 1,
    1)), units = "degrees", template = "geographics")
south <- circular(rep(c(55, 100, 110, 115,
    165, 170, 180, 185, 190, 195, 200, 205,
    210, 215, 220, 225, 230, 235, 240, 245,
    250, 255, 260, 265, 270, 275, 280, 290,
    295, 300, 310), c(1, 1, 1, 1, 1, 2, 1,
    1, 2, 1, 3, 2, 3, 7, 15, 19, 21, 29,
    13, 14, 9, 3, 3, 4, 2, 1, 1, 1, 1, 1,
    1)), units = "degrees", template = "geographics")
```

Return

```r
grid <- circular(seq(0, 360, length.out = 5000),
    units = "degrees", template = "geographics")
asdnorth <- localdepth(x = north, method = "simplicial",
    tau = circular(0))$depth
asdnorthgrid <- localdepth(x = north, y = grid,
    method = "simplicial", tau = circular(0))$depth
asmnorth <- grid[which.max(asdnorthgrid)]
north095 <- north[asdnorth > quantile(asdnorth,
    0.05)]
north095 <- north095[order(north095 - asmnorth)]
snorth095 <- north095[c(1, length(north095))]
```

◂ Return

```r
asdsouth <- localdepth(x = south, method = "simplicial",
    tau = circular(0))$depth
asdsouthgrid <- localdepth(x = south, y = grid,
    method = "simplicial", tau = circular(0))$depth
asmsouth <- grid[which.max(asdsouthgrid)]
south095 <- south[asdsouth > quantile(asdsouth,
    0.05)]
south095 <- south095[order(south095 - asmsouth)]
ssouth095 <- south095[c(1, length(south095))]
```

◂ Return

```
nn <- length(north)
ngamma <- 0
for (i in 1:(nn - 1)) {
    for (j in (i + 1):nn) {
        arcd <- abs(north[i] - north[j])
        ngamma <- ngamma + min(arcd, 360 -
            arcd)
    }
}
ngamma <- 2 * ngamma/(nn * (nn - 1))

ns <- length(south)
sgamma <- 0
for (i in 1:(ns - 1)) {
    for (j in (i + 1):ns) {
        arcd <- abs(south[i] - south[j])
        sgamma <- sgamma + min(arcd, 360 -
                arcd)
        }
}
sgamma <- 2 * sgamma/(ns * (ns - 1))
```

◂ Return

```r
pinfo <- plot(north, stack = TRUE, bins = 75,
    shrink = 1.5, main = "Bird Migration Data - North")
lines(grid, asdnorthgrid, join = TRUE, plot.info = pinfo)
arrows.circular(c(asmnorth, snorth095), length = 0.1,
    lty = c("solid", "dotted", "dotted"))
pinfo <- plot(south, stack = TRUE, bins = 80,
    shrink = 1.5, main = "Bird Migration Data - South")
lines(grid, asdsouthgrid, join = TRUE, plot.info = pinfo)
arrows.circular(c(asmsouth, ssouth095), length = 0.1,
    lty = c("solid", "dotted", "dotted"))
```

◂ Return

```
asdn <- localdepth(x = north, y = c(north,
    south), method = "simplicial", tau = circular(0))$depth
asds <- localdepth(x = south, y = c(north,
    south), method = "simplicial", tau = circular(0))$depth
plot(asdn, asds, type = "p", pch = 20, xlim = c(0,
    0.6), ylim = c(0, 0.6), col = c(rep(1,
    length(north)), rep(2, length(south))),
    cex = c(rep(2, length(north)), rep(1,
        length(south))), xlab = "ASD of pooled sample wrt N
    ylab = "ASD of pooled sample wrt South")
abline(a = 0, b = 1, lty = "dashed")
legend("bottomright", legend = c("North",
    "South"), col = c(1, 2), pch = 20, inset = 0.01)
```

◂ Return

V. Barnett and T. Lewis. *Outliers in Statistical Data*. Wiley, New York, 3rd edition, 1994.

D. Collett. Outliers in circular data. *Applied Statistics*, 29(1): 50–57, 1980.

N. Cressie and T.R.C. Read. *Cressie–Read Statistic*, pages 37–39. Wiley, 1988. In: Encyclopedia of Statistical Sciences, Supplementary Volume, edited by S. Kotz and N.L. Johnson.

T. Fransson. To analyse recoveries in a national atlas - examples from the Swedish project. *Ardea*, 89:21–30, 2001.

F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, and W.A. Stahel. *Robust Statistics: The Approach based on Influence Functions*. Wiley, 1986.

D. Hawkins. *Identification of Outliers*. Chapman & Hall, 1980.

X. He. Robust statistics of directional data: A survey. *Nonparametric Statistics and Related Topics*, pages 87–95, 1992.

R.V. Lenth. Robust measures of location for directional data. *Technometrics*, 23(2):77–81, 1981.

B.G. Lindsay. Efficiency versus robustness: The case for minimum hellinger distance and related methods. *Annals of Statistics*, 22: 1018–1114, 1994.

M. Markatou, A. Basu, and B.G. Lindsay. Weighted likelihood estimating equations: The continuous case. Technical report, Department of Statistics, Columbia University, New York, 1995.

A. SenGupta and A.K. Laha. The slippage problem for the circular normal distribution. *Australian and New Zealand Journal of Statistics*, 43(4):461–471, 2001.

G. Upton. Outliers in circular data. *Journal of Applied Statistics*, 20:229–235, 1993.

These slides are prepared using LaTeX, beamer class and `knitr` package in R . They are compiled with R ver. 3.0.2 running under OS linux-gnu and packages `circular` ver. 0.4-8, `wle` ver. 0.9-8, `localdepth` ver. localdepth and `xtable` ver. 1.7-1.