

Package: rWind (via r-universe)

October 18, 2026

Encoding UTF-8

Type Package

Title Download, Edit and Include Wind and Sea Currents Data in Ecological and Evolutionary Analysis

Version 1.2.0

BugReports <https://github.com/jabiologo/rWind/issues>

Maintainer Javier Fernández-López <jflopez.bio@gmail.com>

Description Downloads and processes surface wind data from the Global Forecast System

<<https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast>>
and sea-current data from Ocean Surface Current Analyses
Real-time (OSCAR). Converts flow speed and direction into
spatial rasters and directed movement costs and conductances
for connectivity and least-cost path analyses.

URL <https://github.com/jabiologo/rWind>,
<https://allthiswasfield.blogspot.com/>

License GPL (>= 3)

LazyData true

Imports raster (>= 2.5-8), terra, gdistance, Matrix, lubridate

Suggests testthat, rmarkdown, knitr, curl, shape

VignetteBuilder knitr

Depends R (>= 3.5)

RoxygenNote 7.3.3

NeedsCompilation no

Author Javier Fernández-López [aut, cre], Klaus Schliep [aut], Yurena Arjona [aut]

Repository <https://cran.r-universe.dev>

Date/Publication 2026-10-18 16:30:02 UTC

RemoteUrl <https://github.com/cran/rWind>

RemoteRef HEAD

RemoteSha 87b07be59d45410933d46f3faf093e1327b32f40

Contents

rWind-package	2
arrowDir	3
cost.active	4
cost.FMGS	5
seaOscar.dl	7
tidy	9
uv2ds	10
wind.data	11
wind.dl	12
wind.dl_2	14
wind.mean	16
wind.series	17
wind2raster	18
Index	19

rWind-package	<i>Download, edit and include wind and sea currents data in ecological and evolutionary analysis</i>
---------------	--

Description

rWind contains tools for downloading, editing and transforming wind and sea currents data from the Global Forecast System (GFS) and Ocean Surface Current Analyses Real-time (OSCAR). It also allows wind and sea currents data to be used to compute minimum-cost paths from flow speed and direction and to perform connectivity analysis.

Details

The complete list of functions can be displayed with `library(help = rWind)`. For more information, please check: <https://allthiswasfield.blogspot.com/>.

Author(s)

Javier Fernández-López
Klaus Schliep
Yurena Arjona
Maintainer: Javier Fernández-López jflopez.bio@gmail.com

See Also

- Useful links:
- <https://github.com/jabiologo/rWind>
 - <https://allthiswasfield.blogspot.com/>
 - Report bugs at <https://github.com/jabiologo/rWind/issues>

arrowDir	<i>Arrow direction fitting for Arrowhead function from "shape" package</i>
----------	--

Description

arrowDir adapts wind direction value to be used by Arrowhead function from "shape" package to plot wind direction for each coordinate.

Usage

```
arrowDir(W)
```

Arguments

W	An object of class rWind or a data.frame which should content a column named "dir".
---	---

Details

Angle argument of Arrowhead function from "shape" package needs to be fed in an anti-clockwise way, relative to x-axis, in degrees 0,360. arrowDir function adapts wind direction provided by wind.fit (clockwise, relative to y-axis) to requirements of Arrowhead.

Value

A numeric vector with one angle per grid cell, in degrees anticlockwise from the positive x-axis, for plotting with `shape::Arrowhead()`. For an `rWind_series`, only the first time step is used.

Note

arrowDir function works always together with Arrowhead function from "shape" package.

Author(s)

Javier Fernández-López

References

Karline Soetaert (2017). shape: Functions for Plotting Graphical Shapes, Colors. R package version 1.4.3. <https://CRAN.R-project.org/package=shape>

See Also

[wind.dl](#)

Examples

```
data(wind.data)

# Create a vector with wind direction (angles) adapted
alpha <- arrowDir(wind.data)
# Plot arrows if the optional shape package is available
if (requireNamespace("shape", quietly = TRUE)) {
  plot(wind.data$lon, wind.data$lat, type = "n")
  shape::Arrowhead(wind.data$lon, wind.data$lat,
    angle = alpha,
    arr.length = 0.1, arr.type = "curved"
  )
}
```

cost.active	<i>Relative cost of active movement through a flow</i>
-------------	--

Description

cost.active() is a deliberately simple, general index based on signed flow support. The flow component parallel to movement is $\text{speed} * \cos(\text{angle})$. Supportive flow lowers cost, opposing flow raises it, and perpendicular flow has no effect. This is a relative index, not physical travel time or energy.

Usage

```
cost.active(flow.direction, flow.speed, target, speed.scale = NULL)
```

Arguments

flow.direction	Flow direction in degrees, pointing towards where the flow moves.
flow.speed	Non-negative flow speed.
target	Movement direction in degrees.
speed.scale	Positive reference speed used to make flow support dimensionless. By default, the median positive input speed is used.

Details

Let $z = \text{speed} * \cos(\text{angle}) / \text{speed.scale}$. When $z \geq 0$, cost is $1 / (1 + z)$; when $z < 0$, cost is $1 - z$. Thus support equal to the reference speed halves cost, opposition equal to the reference speed doubles it, and $\text{cost}(-z) = 1 / \text{cost}(z)$.

Value

A numeric vector of relative movement costs, or a numeric array when the input supplying the output shape is an array. Scalar inputs are recycled to match the longer input. A value below one indicates support, one indicates calm or perpendicular flow, and a value above one indicates opposition. Invalid flow values give Inf, except that zero speed gives a cost of one even when direction is missing.

cost.FMGS

*Compute flow-based cost or conductance***Description**

flow.dispersion computes movement conductance through a flow such as sea or wind currents. Passive movement uses a restricted version of the formula described in Felícisimo et al. (2008):

Usage

```
cost.FMGS(
  wind.direction,
  wind.speed,
  target,
  type = c("passive", "active"),
  speed.scale = NULL
)

flow.dispersion(
  x,
  fun = cost.FMGS,
  output = "transitionLayer",
  type = c("passive", "active"),
  speed.scale = NULL,
  ...
)
```

Arguments

wind.direction	A vector or scalar containing wind directions.
wind.speed	A vector or scalar containing wind speeds.
target	direction of the target cell
type	Movement model. In "passive" mode, deviations of 90 degrees or more from the flow direction are forbidden. In "active" mode, supportive flow lowers relative cost and opposing flow raises it; movement against the flow remains possible.
speed.scale	Positive reference speed for the active model. If NULL, the median positive speed of the complete raster layer is used. For comparisons among dates or regions, supply one fixed value in the same units as wind.speed.

x	A <code>SpatRaster</code> returned by <code>wind2raster()</code> with direction and speed layers, a legacy <code>RasterStack</code> , or a list of either type.
fun	A function to compute the cost to move between cells. The default is <code>cost.FMGS</code> from Felicísimo et al. (2008), see details.
output	This argument allows to select different kinds of output. "raw" mode creates a matrix (class " <code>dgCMatrix</code> ") with transition costs between all cells in the raster. "transitionLayer" creates a <code>TransitionLayer</code> object with conductance values to be used with "gdistance" package.
...	Further arguments passed to or from other methods.

Details

$\text{Cost} = (1/\text{Speed}) * (\text{HorizontalFactor})$

where `HorizontalFactor` incrementally penalizes angular deviations from the flow direction and prevents passive movement at deviations of 90 degrees or more. Active movement uses a general relative index based on signed flow support rather than the original formula.

Value

`flow.dispersion()` returns a `gdistance::TransitionLayer` for `output = "transitionLayer"`, with directed conductances ($1/\text{cost}$) between the eight neighbouring cells, or a sparse matrix of class `dgCMatrix` containing edge costs for `output = "raw"`. Rows index origins and columns index destinations. Costs average local costs at both cells and are multiplied by $\sqrt{2}$ for diagonal moves. In the raw matrix, `Inf` marks forbidden neighbouring moves; zeros represent absent edges, not free moves. In the conductance matrix, forbidden moves have zero conductance. A list input returns a list of the corresponding output objects.

`cost.FMGS()` returns numeric local movement costs for the supplied flow values and target direction (a vector, or a matrix for matrix inputs). Passive costs are the horizontal factor divided by speed, with `Inf` for forbidden moves. Active costs follow `cost.active()`.

Note

The active model is a relative connectivity index, not a mechanistic model of animal movement. It ignores the component of flow perpendicular to movement.

Costs are relative per grid step. The diagonal correction assumes a square grid, but no correction is made for the changing physical distances of longitude/latitude cells. Do not interpret results as physical time, energy, or distance or compare them directly across grid resolutions or distant latitude ranges.

Note that for large data sets, it could take a while. For large study areas is strongly advised perform the analysis in a remote computer or a cluster.

Author(s)

Javier Fernández-López; Klaus Schliep; Yurena Arjona

References

Felicísimo, Á. M., Muñoz, J., & González-Solis, J. (2008). Ocean surface winds drive dynamics of transoceanic aerial movements. *PLoS One*, 3(8), e2928.

Jacob van Etten (2017). R Package gdistance: Distances and Routes on Geographical Grids. *Journal of Statistical Software*, 76(13), 1-21. doi:10.18637/jss.v076.i13

See Also

[wind.dl](#), [wind2raster](#)

Examples

```
require(gdistance)

data(wind.data)

wind <- wind2raster(wind.data)

Conductance <- flow.dispersion(wind, type = "passive")

conductance_matrix <- gdistance::transitionMatrix(Conductance)
conductance_matrix[1:6, 1:6]
Matrix::image(conductance_matrix)
```

sea0scar.dl

OSCAR Sea currents data download

Description

seaOscar.dl downloads sea currents data from the Ocean Surface Current Analyses Real-time (OSCAR) (PO.DAAC, doi:10.5067/OSCAR03D01). Geospatial resolution is 0.33 degrees and sea currents are calculated for 15 m depth. Longitudes in the usual -180 to 180 notation are translated to the 20 to 420 degree axis used by the service.

Usage

```
sea0scar.dl(
  yyyy,
  mm,
  dd,
  lon1,
  lon2,
  lat1,
  lat2,
  type = "read-data",
  trace = 1
)
```

Arguments

yyyy	Selected year.
mm	Selected month.
dd	Selected day.
lon1	Western longitude
lon2	Eastern longitude
lat1	Northern latitude
lat2	Southern latitude
type	Output type. "read-data" is selected by default, creating an R object. If you choose "csv", seaOscar.dl create a CSV file in your working directory named "oscar_yyyy_mm_dd.csv".
trace	if trace = 1 (by default) track downloaded files

Details

The output type is determined by type="csv" or type="read-data". If type="csv" is selected, the function creates an "oscar_yyyy_mm_dd.csv" file that is downloaded at the work directory. If type="read-data" is selected, an R object (data.frame) is created.

Value

An object of classes `rWind` and `data.frame`, with one row per grid cell and columns `time` (POSIXct, UTC), `lat` and `lon` (degrees), `u` and `v` (eastward and northward current components, m/s), `dir` (direction of movement, in degrees clockwise from north), and `speed` (m/s). These values describe currents at the OSCAR reference depth. With type = "csv", a CSV file is also written in the working directory and the same object is returned.

Note

The NOAA ERDDAP endpoint used here provides a historical subset of OSCAR, not its full temporal coverage. Check the available dates in the [dataset metadata](#) before requesting data. Requests outside that coverage fail.

Author(s)

Javier Fernández-López (jflopez.bio@gmail.com)

References

[doi:10.5067/OSCAR03D01](https://doi.org/10.5067/OSCAR03D01)

See Also

[wind.dl_2](#), [wind2raster](#)

Examples

```
# Download historical sea currents near the Galapagos Islands
```

```
seaOscar.dl(2014, 9, 26, -93, -88, 2, -3)
```

tidy

Transforming a rWind_series object into a data.frame

Description

The output of tidy is always a data.frame. It is therefore suited for further manipulation by packages like dplyr, reshape2, ggplot2 and ggvis.

Usage

```
tidy(x, ...)

## S3 method for class 'rWind_series'
tidy(x, ...)
```

Arguments

```
x          An object to be converted into a tidy data.frame
...        extra arguments
```

Value

For an rWind_series, an object of classes rWind and data.frame, formed by stacking the time steps in their original order. Each row represents one grid cell at one time. The seven columns are time (POSIXct, UTC), lat and lon (numeric coordinates in degrees), ugrd10m and vgrd10m (numeric eastward and northward wind components in m/s), dir (numeric direction of movement, in degrees clockwise from north), and speed (numeric wind speed in m/s). Values are retained for each time step; no temporal averaging is performed.

Examples

```
data(wind.series)
df <- tidy(wind.series)
head(df)
# Compute mean speed at each location using the combined data frame
mean_speed <- stats::aggregate(speed ~ lat + lon, data = df, FUN = mean)
head(mean_speed)
```

uv2ds

*Transform U and V components in direction and speed and vice versa***Description**

Transform U and V components in direction and speed and vice versa

Usage

```
uv2ds(u, v)
```

```
ds2uv(d, s)
```

Arguments

u	U component.
v	U component.
d	direction (degrees).
s	speed (m/s).

Value

uv2ds() returns a numeric matrix with columns dir (degrees clockwise from north, towards where the flow moves) and speed (in the same units as U and V), with one row per input pair.

ds2uv() returns a numeric matrix with columns u (eastward component) and v (northward component), in the same units as s, with one row per direction-speed pair.

Note

Multiple U and V values can be processed. "dir" denotes where the wind/sea current is going (toward), not from where is coming.

Author(s)

Javier Fernández-López (jflopez.bio@gmail.com)

See Also

[wind.mean](#), [wind2raster](#)

Examples

```
(ds <- uv2ds(c(1, 1, 3, 1), c(1, 1.7, 3, 1)))
ds2uv(ds[, 1], ds[, 2])
```

wind.data*Wind data example*

Description

This is an example of wind data obtained with `wind.dl` function for the Iberian Peninsula coordinates on 12/February/2015 at 00:00 (UTC)

Format

A list with one `data.frame` with 651 observations on the following 7 variables:

`list("time (UTC)")` a numeric with selected time of wind data

`list("latitude (degrees_north)")` a numeric with latitude values

`list("longitude (degrees_east)")` a numeric with longitude values

`list("ugrd10m (m s-1)")` a numeric with U component of wind data

`list("vgrd10m (m s-1)")` a numeric with V component of wind data

`list("dir")` a numeric with direction of wind data

`list("speed")` a numeric with speed of wind data

Details

This data set is the result of:

```
wind.data <- wind.dl(2015,2,12,0,-10,5,35,45)
```

Source

<https://allthiswasfield.blogspot.com/2016/12/rwind-r-package-released.html>

References

<https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast>

Examples

```
data(wind.data)
str(wind.data)
head(wind.data[[1]])
```

wind.dl

*Wind-data download***Description**

wind.dl downloads wind data from the Global Forecast System (GFS) of the USA's National Weather Service (NWS) (<https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast>). Wind data are taken from NOAA/NCEP Global Forecast System (GFS) Atmospheric Model collection. Geospatial resolution is 0.5 degrees (approximately 50 km), and wind is calculated for Earth surface, at 10 m. More metadata information for the current PacIOOS data: https://pae-paha.pacioos.hawaii.edu/erddap/griddap/ncep_global.graph Historical data are obtained from the official NOAA/NCEI GFS 0.5 degree archive: <https://www.ncei.noaa.gov/access/metadata/landing-page/bin/iso?id=gov.noaa.ncdc:C00634>

Usage

```
wind.dl(
  yyyy,
  mm,
  dd,
  tt,
  lon1,
  lon2,
  lat1,
  lat2,
  type = "read-data",
  trace = 1,
  source = c("auto", "pacioos", "ncei")
)

read.rWind(file)
```

Arguments

yyyy	Selected year.
mm	Selected month.
dd	Selected day.
tt	Selected time. There are currently several options at the GFS database: 00:00 - 03:00 - 06:00 - 09:00 - 12:00 - 15:00 - 18:00 - 21:00 (UTC).
lon1	Western longitude
lon2	Eastern longitude
lat1	Southern latitude
lat2	Northern latitude

type	Output type. "read-data" is selected by default, creating an R object. If you choose "csv", wind.dl create a a CSV file in your working directory named "wind_yyyy_mm_dd_tt.csv".
trace	if trace = 1 (by default) track downloaded files
source	Data provider. "auto" selects PacIOOS for dates in its current coverage and NOAA/NCEI for older dates. Use "pacioos" or "ncei" to force one provider.
file	file name of the saved ".csv" files.

Details

The output type is determined by type="csv" or type="read-data". If type="csv" is selected, the function creates a "wind_yyyy_mm_dd_tt.csv" file that is downloaded at the work directory. If type="read-data" is selected, an R object (data.frame) is created.

Value

wind.dl() and read.rWind() return an object of classes rWind and data.frame, with one row per grid cell. Columns are time (POSIXct, UTC), lat and lon (degrees), ugrd10m and vgrd10m (eastward and northward wind components in m/s), dir (direction of movement, in degrees clockwise from north), and speed (m/s). With type = "csv", wind.dl() also writes a CSV file in the working directory and returns the same object.

Note

Longitude coordinate are provided by GFS dataset in 0/360 notation and transformed internally into -180/180. Wind "dir" denotes where the wind is going (toward), not from where is coming. With source = "auto", dates before the current PacIOOS coverage are requested from NOAA/NCEI.

Author(s)

Javier Fernández-López (jflopez.bio@gmail.com)

References

<http://www.digital-geography.com/cloud-gis-getting-weather-data/#.WDOWmbV1DCL>
https://pae-paha.pacioos.hawaii.edu/erddap/griddap/ncep_global.graph

See Also

[wind.dl_2](#), [wind2raster](#)

Examples

```
# Download wind for Iberian Peninsula region at 2015, February 12, 00:00
```

```
wind.dl(2015, 2, 12, 0, -10, 5, 35, 45)
```

wind.dl_2

*Wind-data download***Description**

wind.dl_2 downloads time-series wind data from the Global Forecast System (GFS) of the USA's National Weather Service (NWS) (<https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast>). Wind data are taken from NOAA/NCEP Global Forecast System (GFS) Atmospheric Model collection. Geospatial resolution is 0.5 degrees (approximately 50 km), and wind is calculated for Earth surface, at 10 m. More metadata information for current data: <https://pae-paha.pacioos.hawaii.edu/erddap/grid>. Historical data are obtained from the official NOAA/NCEI GFS 0.5 degree archive.

Usage

```
wind.dl_2(
  time,
  lon1,
  lon2,
  lat1,
  lat2,
  type = "read-data",
  trace = 1,
  source = c("auto", "pacioos", "ncei")
)

## S3 method for class 'rWind_series'
x[[i, exact = TRUE]]
```

Arguments

time	a scalar or vector of POSIXt or Date objects or an character which can transformed into those, see example below. There are currently these options at the GFS database for the hours: 00:00 - 03:00 - 06:00 - 09:00 - 12:00 - 15:00 - 18:00 - 21:00 (UTC) (TO).
lon1	Western longitude
lon2	Eastern longitude
lat1	Southern latitude
lat2	Northern latitude
type	Output type. "read-data" is selected by default, creating an R object. If you choose "csv", wind.dl create a a CSV file in your work directory named "wind_yyyy_mm_dd_tt.csv".
trace	if trace = 1 (by default) track downloaded files
source	Data provider. "auto" selects PacIOOS for dates in its current coverage and NOAA/NCEI for older dates. Use "pacioos" or "ncei" to force one provider.
x	object from which to extract element(s).
i	indices specifying elements to extract.
exact	Controls possible partial matching (not used yet).

Details

To get the same format as `wind.dl`, you should run `tidy` function from `wind.dl_2` output. The output type is determined by `type="csv"` or `type="read-data"`. If `type="csv"` is selected, the function creates a `"wind_yyyy_mm_dd_tt.csv"` file that is downloaded at the work directory. If `type="read-data"` is selected, an `rWind_series` object is created.

Value

With `type = "read-data"`, `wind.dl_2()` returns an object of classes `rWind_series` and `list`, with one element per requested time. Each element stores a data frame of U and V wind components (m/s); element names encode UTC times and the `lat_lon` attribute stores the shared grid coordinates in degrees. Extracting a time step with `[[` returns an object of classes `rWind` and `data.frame`, with columns `time`, `lat`, `lon`, `ugrd10m`, `vgrd10m`, `dir`, and `speed`, as described in [wind.dl\(\)](#). With `type = "csv"`, `wind.dl_2()` writes one CSV file per time step in the working directory and returns `NULL` invisibly.

Note

`wind.dl_2` requires two dates that represent the boundaries of the time lapse to download wind series data. U and V vector components allow you to create wind averages or tendencies for each coordinate at the study area. Longitude coordinates are provided by GFS dataset in 0/360 notation and transformed internally into -180/180. "dir" denotes where the wind/sea current is going (toward), not from where is coming. With `source = "auto"`, dates before the current PacIOOS coverage are requested from NOAA/NCEI.

Author(s)

Javier Fernández-López (jflopez.bio@gmail.com)

References

<http://www.digital-geography.com/cloud-gis-getting-weather-data/#.WDOWmbV1DCL>

https://pae-paha.pacioos.hawaii.edu/erddap/griddap/ncep_global.graph

See Also

[wind.mean](#), [wind2raster](#), [wind.dl](#), [as_datetime](#), [as.POSIXct](#)

Examples

```
# Download two time steps for a small region in southern Iberia

dt <- as.POSIXct(c("2015-02-12 09:00:00", "2015-02-12 12:00:00"),
  tz = "UTC")
ww <- wind.dl_2(dt, -7, -4, 34.5, 37.5)
tidy(ww)
```

wind.mean

Wind-data mean

Description

wind.mean computes the mean (average) wind speed and wind direction of a time series dataset of winds of the same region. Summaries of time series are not trivial to compute. We compute the arithmetic mean for the wind speed. The direction as the circular mean, see https://en.wikipedia.org/wiki/Circular_mean for more details. The U and V components are afterwards transformed from these values.

Usage

```
wind.mean(x)
```

Arguments

x An object of class rWind_series

Value

An object of classes rWind and data.frame, with one row per grid cell and columns time, lat, lon, ugrd10m, vgrd10m, dir, and speed. Speed is the arithmetic mean across time steps (m/s), and direction is the circular mean (degrees clockwise from north, towards where the wind moves). U and V (m/s) are derived from these mean values. Coordinates are in degrees, and time records the first time step (UTC).

Note

For large time series, it could take a while.

Author(s)

Javier Fernández-López (jflopez.bio@gmail.com)

References

https://en.wikipedia.org/wiki/Cross_product

See Also

[wind.dl](#)

Examples

```
data(wind.series)
wind_average <- wind.mean(wind.series)
```

wind.series*Wind series example*

Description

This is an example of a wind series data obtained with `wind.dl` function for New Zealand area on 3/January/2015 at all the available times: 00:00 - 03:00 - 06:00 - 09:00 - 12:00 - 15:00 - 18:00 - 21:00 (UTC)

Format

The format is an `rWind` list of 8 `data.frame`. Each `data.frame` contain 961 observations on the following 7 variables:

list("time (UTC)") a factor with selected time of wind data

list("latitude (degrees_north)") a factor with latitude values

list("longitude (degrees_east)") a factor with longitude values

list("ugrd10m (m s-1)") a factor with U component of wind data

list("vgrd10m (m s-1)") a factor with V component of wind data

list("dir") a numeric with direction of wind data

list("speed") a numeric with speed of wind data

Details

This data set is the result of:

```
library(lubridate) dt <- seq(ymd_h(paste(2015,1,3,00, sep="-")), ymd_h(paste(2015,1,3,21, sep="-")), by="3 hours")
wind.series <- wind.dl_2(dt, 164, 179, -48, -33)
```

Source

<https://allthiswasfield.blogspot.com/2016/12/rwind-r-package-released.html>

References

<https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast>

Examples

```
data(wind.series)
str(tidy(wind.series))
```

wind2raster*Wind data to terra rasters*

Description

wind2raster() converts an rWind object to a two-layer [terra::SpatRaster](#) containing wind direction and speed. An rWind_series is converted to a list of SpatRaster objects, one per time step. All outputs use WGS 84 (EPSG:4326).

Usage

```
wind2raster(x)
```

Arguments

x An object of class rWind or rWind_series.

Value

A [terra::SpatRaster](#), or a list of them, with layers direction and speed.

Author(s)

Javier Fernández-López (jflopez.bio@gmail.com)

See Also

[wind.dl](#)

Examples

```
data(wind.data)

# Create a SpatRaster with wind direction and speed layers

wind2raster(wind.data)
```

Index

- * **~anisotropy**
 - cost.FMGS, 5
- * **~average**
 - wind.mean, 16
- * **~conductance**
 - cost.FMGS, 5
- * **~currents**
 - seaOscar.dl, 7
- * **~gfs**
 - wind.dl, 12
 - wind.dl_2, 14
 - wind2raster, 18
- * **~mean**
 - wind.mean, 16
- * **~sea**
 - seaOscar.dl, 7
- * **~wind**
 - arrowDir, 3
 - uv2ds, 10
 - wind.dl, 12
 - wind.dl_2, 14
 - wind2raster, 18
- * **datasets**
 - wind.data, 11
 - wind.series, 17
- * **package**
 - rWind-package, 2
- [[.rWind_series(wind.dl_2), 14
- arrowDir, 3
- as.POSIXct, 15
- as_datetime, 15
- cost.active, 4
- cost.active(), 6
- cost.FMGS, 5
- ds2uv(uv2ds), 10
- flow.dispersion(cost.FMGS), 5
- read.rWind(wind.dl), 12
- rWind(rWind-package), 2
- rWind-package, 2
- seaOscar.dl, 7
- terra::SpatRaster, 18
- tidy, 9
- uv2ds, 10
- wind.data, 11
- wind.dl, 3, 7, 12, 15, 16, 18
- wind.dl(), 15
- wind.dl_2, 8, 13, 14
- wind.mean, 10, 15, 16
- wind.series, 17
- wind2raster, 7, 8, 10, 13, 15, 18