Package: raytracing (via r-universe)

August 21, 2024

Title Rossby Wave Ray Tracing

Version 0.5.0

Date 2020-10-22

Description Rossby wave ray paths are traced from a determined source, specified wavenumber, and direction of propagation. ``raytracing'' also works with a set of experiments changing these parameters, making possible the identification of Rossby wave sources automatically. The theory used here is based on classical studies, such as Hoskins and Karoly (1981) <doi:10.1175/1520-0469(1981)038%3C1179:TSLROA%3E2.0.CO;2>, Karoly (1983) <doi:10.1016/0377-0265(83)90013-1>, Hoskins and Ambrizzi (1993) <doi:10.1175/1520-0469(1993)050%3C1661:RWPOAR%3E2.0.CO;2>, and Yang and Hoskins (1996) <doi:10.1175/1520-0469(1996)053%3C2365:PORWON%3E2.0.CO;2>.

License GPL-3

Encoding UTF-8

LazyData no

Imports ncdf4, graphics, sf, units, utils

Suggests testthat, covr, lwgeom

URL `https://github.com/salvatirehbein/raytracing/`

BugReports `https://github.com/salvatirehbein/raytracing/issues/`

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Depends R (>= 3.5.0)

Repository https://salvatirehbein.r-universe.dev

RemoteUrl https://github.com/salvatirehbein/raytracing

RemoteRef HEAD

RemoteSha f5775f164e4d00476438c80dca9f1c0890c86b7b

Contents

`betaks` *Calculates Beta and Ks*

Description

`betaks` ingests the time-mean zonal wind (u), transform it in mercator coordinates (um); calculates the meridional gradient of the absolute vorticity (beta) in mercator coordinates (betam); and, finally, calculates stationary wavenumber (Ks) in mercator coordinates (ksm) (see: Hoskins and Ambrizzi, 1993). `betaks` returns the um, betam, and lat, for being ingested in `ray` or `ray_source`.

Usage

```
betaks(
  u,
  lat = "lat",
  lon = "lon",
  uname = "uwnd",
  ofile,
  a = 6371000,
  plots = FALSE,
  show.warnings = FALSE
)
```

Arguments

`u` String indicating the input data filename. The file to be passed consists in a netCDF file with only time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns). **u** also can be a numerical matrix with time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns).

`lat` String indicating the name of the latitude field. If **u** is a matrix, **lat** must be numeric.

`lon` String indicating the name of the longitude field.If **u** is a matrix, **lon** must be numeric from 0 to 360.

`uname` String indicating the variable name field

`ofile` String indicating the file name for store output data. If missing, will not return a netCDF file

`a` Numeric indicating the Earth's radio (m)

`plots` Logical, if TRUE will produce filled.countour plots

`show.warnings` Logical, if TRUE will warns about NaNs in sqrt(<0)

Value

list with one vector (lat) and 3 matrices (um, betam, and ksm)

Examples

```
{
# u is NetCDF and lat and lon characters
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
b <- betaks(u = input, plots = TRUE)
b$ksm[] <- ifelse(b$ksm[] >= 16 |
                  b$ksm[] <= 0, NA, b$ksm[])
cores <- c("#ff0000","#ff5a00","#ff9a00","#ffce00","#f0ff00")
graphics::filled.contour(b$ksm[, -c(1:5, 69:73)] ,
                         col = rev(colorRampPalette(cores, bias = 0.5)(20)),
                         main = "Ks")

# u, lat and lon as numeric
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.bin",
                      package = "raytracing")
u <- readBin(input,
             what = numeric(),
             size = 4,
             n = 144*73*4)
lat <- seq(-90, 90, 2.5)
lon <- seq(-180, 180 - 1, 2.5)
u <- matrix(u,
            nrow = length(lon),
            ncol = length(lat))
graphics::filled.contour(u, main = "Zonal Wind Speed [m/s]")
b <- betaks(u, lat, lon)
b$ksm[] <- ifelse(b$ksm[] >= 16 |
                  b$ksm[] <= 0, NA, b$ksm[])
cores <- c("#ff0000","#ff5a00","#ff9a00","#ffce00","#f0ff00")
graphics::filled.contour(b$ksm[, -c(1:5, 69:73)] ,
                         col = rev(colorRampPalette(cores, bias = 0.5)(20)),
                         main = "Ks")
}
```

betam *Calculates Meridional Gradient of the Absolute Vorticity (beta) in mercator coordinates (betam)*

Description

`betam` ingests the time-mean zonal wind (u), transform it in mercator coordinates (um) and then calculates the meridional gradient of the absolute vorticity (beta) in mercator coordinates (betam) using equation Karoly (1983). `betam` returns a list with the **u**, **betam**, and **lat** for being ingested in `Ktotal`, `Ks`, `ray` or `ray_source`.

Usage

```
betam(
  u,
  lat = "lat",
  lon = "lon",
  uname = "uwnd",
  ofile,
  a = 6371000,
  plots = FALSE,
  show.warnings = FALSE
)
```

Arguments

`u` String indicating the input data filename. The file to be passed consists in a netCDF file with only time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns). **u** also can be a numerical matrix with time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns).

`lat` String indicating the name of the latitude field. If **u** is a matrix, **lat** must be numeric.

`lon` String indicating the name of the longitude field.If **u** is a matrix, **lon** must be numeric from 0 to 360.

`uname` String indicating the variable name field

`ofile` String indicating the file name for store output data. If missing, it will not return a netCDF file

`a`	Numeric indicating the Earth’s radio (m)
`plots`	Logical, if TRUE will produce filled.countour plots
`show.warnings`	Logical, if TRUE will warns about NaNs in sqrt(<0)

Value

list with one vector (lat) and 2 matrices (u and betam)

Examples

```
{
# u is NetCDF and lat and lon characters
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
b <- betam(u = input, plots = TRUE)
cores <- c("#ff0000","#ff5a00","#ff9a00","#ffce00","#f0ff00")
graphics::filled.contour(b$betam/10e-12,
                         zlim = c(0, 11),
                         col = rev(colorRampPalette(cores)(24)),
                         main = "Beta Mercator (*10e-11)")
# u, lat and lon as numeric
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.bin",
                      package = "raytracing")
u <- readBin(input,
             what = numeric(),
             size = 4,
             n = 144*73*4)
lat <- seq(-90, 90, 2.5)
lon <- seq(-180, 180 - 1, 2.5)
u <- matrix(u,
            nrow = length(lon),
            ncol = length(lat))
graphics::filled.contour(u, main = "Zonal Wind Speed [m/s]")
}
```

`coastlines` *Coastlines*

Description

Geometry of coastlines, class "sfc_MULTILINESTRING" "sfc" from the package "sf"

Usage

```
data(coastlines)
```

Format

Geometry of coastlines "sfc_MULTILINESTRING"

Geometry of coastlines "sfc_MULTILINESTRING" data(coastlines)

Source

https://www.naturalearthdata.com/downloads/10m-physical-vectors/10m-coastline/

Ks *Calculates Total Wavenumber for Stationary Rossby Waves (Ks)*

Description

Ks ingests the time-mean zonal wind (u) and calculates the Total Wavenumber for Stationary Rossby waves (Ks) in mercator coordinates (see: Hoskins and Ambrizzi, 1993). Stationary Rossby waves are found when zonal wave number (k) is constant along the trajectory, which leads to wave frequency (omega) zero. In this code Ks is used to distinguish the total wavenumber for Stationary Rossby Waves (Ks) from the total wavenumber for Rossby waves (K), and zonal wave number (k). Ks returns a list with Ks in mercator coordinates (ksm).

Usage

```
Ks(
  u,
  lat = "lat",
  lon = "lon",
  uname = "uwnd",
  ofile,
  a = 6371000,
  plots = FALSE,
  show.warnings = FALSE
)
```

Arguments

u	String indicating the input data filename. The file to be passed consists in a netCDF file with only time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns). **u** also can be a numerical matrix with time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns).
lat	String indicating the name of the latitude field. If **u** is a matrix, **lat** must be numeric.

`lon` String indicating the name of the longitude field.If **u** is a matrix, **lon** must be numeric from 0 to 360.

`uname` String indicating the variable name field

`ofile` String indicating the file name for store output data. If missing, will not return a netCDF file

`a` Numeric indicating the Earth's radio (m)

`plots` Logical, if TRUE will produce filled.countour plots

`show.warnings` Logical, if TRUE will warns about NaNs in sqrt(<0)

Value

list with one vector (lat) and 1 matrix (Ksm)

Examples

```
{
# u is NetCDF and lat and lon characters
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
Ks <- Ks(u = input, plots = TRUE)
Ks$ksm[] <- ifelse(Ks$ksm[] >= 16 |
                   Ks$ksm[] <= 0, NA, Ks$ksm[])
cores <- c("#ff0000","#ff5a00","#ff9a00","#ffce00","#f0ff00")
graphics::filled.contour(Ks$ksm[, -c(1:5, 69:73)] ,
                         col = rev(colorRampPalette(cores, bias = 0.5)(20)),
                         main = "Ks")
}
```

`Ktotal` *Calculates Total Wavenumber for Rossby Waves (K)*

Description

`Ktotal` ingests the time-mean zonal wind (u) and calculates the Rossby wavenumber (K) (non-zero frequency waves) in mercator coordinates. In this code Ktotal is used to distinguish the total wavenumber (K) from zonal wave number (k). For stationary Rossby Waves, please see `Ks`. `Ktotal` returns a list with K in mercator coordinates (ktotal_m).

Usage

```
Ktotal(
  u,
  lat = "lat",
  lon = "lon",
  uname = "uwnd",
  cx,
  ofile,
  a = 6371000,
  plots = FALSE,
  show.warnings = FALSE
)
```

Arguments

`u` String indicating the input data filename. The file to be passed consists in a netCDF file with only time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns). **u** also can be a numerical matrix with time-mean zonal wind at one pressure level, latitude in ascending order (not a requisite), and longitude from 0 to 360. It is required that the read dimensions express longitude (in rows) x latitude (in columns).

`lat` String indicating the name of the latitude field. If **u** is a matrix, **lat** must be numeric

`lon` String indicating the name of the longitude field.If **u** is a matrix, **lon** must be numeric from 0 to 360.

`uname` String indicating the variable name field

`cx` numeric. Indicates the zonal phase speed. Must be greater than zero. For cx equal to zero (stationary waves see `Ks`)

`ofile` String indicating the file name for store output data. If missing, will not return a netCDF file

`a` Numeric indicating the Earth's radio (m)

`plots` Logical, if TRUE will produce filled.countour plots

`show.warnings` Logical, if TRUE will warns about NaNs in sqrt(<0)

Value

list with one vector (lat) and 1 matrix (ktotal_m)

Examples

```
{
# u is NetCDF and lat and lon characters
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
Ktotal <- Ktotal(u = input, cx = 6, plots = TRUE)
cores <- c("#ff0000","#ff5a00","#ff9a00","#ffce00","#f0ff00")
graphics::filled.contour(Ktotal$ktotal_m[, -c(1:5, 69:73)] ,
                         col = rev(colorRampPalette(cores, bias = 0.5)(20)),
                         main = "K")
}
```

ray *Calculates the Rossby waves ray paths*

Description

ray returns the Rossby wave ray paths (lat/lon) triggered from one initial source/position (x0, y0), one total wavenumber (K), and one direction set up when invoking the function. ray must ingest the meridional gradient of the absolute vorticity in mercator coordinates**betam**, the zonal mean wind **u**, and the latitude vector (**lat**). Those variables can be obtained (recommended) using betaks function. The zonal means of the basic state will be calculated along the **ray** program, as well as the conversion to mercator coordinates of **u**.

Usage

```
ray(
  betam,
  u,
  lat,
  x0,
  y0,
  K,
  dt,
  itime,
  direction,
  cx = 0,
  interpolation = "trin",
  tl = 1,
  a = 6371000,
  verbose = FALSE,
  ofile
)
```

Arguments

betam	matrix (longitude = rows x latitude from minor to major = columns) obtained with betaks. **betam** is the meridional gradient of the absolute vorticity in mercator coordinates
u	matrix (longitude = rows x latitude from minor to major = columns) obtained with betaks. Is the zonal wind speed in the appropriate format for the ray. It will be converted in mercator coordinates inside the ray
lat	Numeric vector of latitudes from minor to major (ex: -90 to 90). Obtained with betaks
x0	Numeric value. Initial longitude (choose between -180 to 180)
y0	Numeric value. Initial latitude
K	Numeric value; Total Rossby wavenumber

`dt` Numeric value; Timestep for integration (hours)

`itime` Numeric value; total integration time. For instance, 10 days times 4 times per day

`direction` Numeric value (possibilities: 1 or -1) It controls the wave displacement: If 1, the wave goes to the north of the source; If -1, the wave goes to the south of the source.

`cx` numeric. Indicates the zonal phase speed. The program is designed for eastward propagation (cx > 0) and stationary waves (cx = 0, the default).

`interpolation` Character. Set the interpolation method to be used: `trin` or `ypos`

`tl` Numeric value; Turning latitude. Do not change this! It will always start with a positive tl (1) and automatically change to negative (-1) after the turning latitude

`a` Earth's radio (m)

`verbose` Boolean; if TRUE (default) return messages during compilation

`ofile` Character; Output file name with .csv extension, for instance, "/user/ray.csv"

Value

sf data.frame

See Also

`ray_source`

Examples

```
{
# For Coelho et al. (2015):
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                     package = "raytracing")
b <- betaks(u = input)
rt <- ray(betam = b$betam,
          u = b$u,
          lat = b$lat,
          K = 3,
          itime = 10 * 4,
          x0 = -130,
          y0 = -30,
          dt = 6,
          direction = -1,
          cx = 0,
          interpolation = "trin")
rp <- ray_path(rt$lon, rt$lat)
plot(rp,
     main = "Coelho et al. (2015): JFM/2014",
     axes = TRUE,
     cex = 2,
     graticule = TRUE)
}
```

`raytracing` *raytracing: Rossby Wave Ray Tracing*

Description

Rossby wave ray paths are traced from a determined source, specified wavenumber, and direction of propagation. 'raytracing' also works with a set of experiments changing these parameters, making possible the identification of Rossby wave sources automatically.

Authors

- Amanda Rehbein (ORCID: https://orcid.org/0000-0002-8714-7931 - mantainer: amanda.rehbein@usp.br)
- Tercio Ambrizzi (ORCID: https://orcid.org/0000-0001-8796-7326)
- Sergio Ibarra Espinosa (ORCID: https://orcid.org/0000-0002-3162-1905)
- Livia Marcia Mosso Dutra (ORCID: https://orcid.org/0000-0002-1349-7138)

References

Hoskins, B. J., & Ambrizzi, T. (1993). Rossby wave propagation on a realistic longitudinally varying flow. Journal of the Atmospheric Sciences, 50(12), 1661-1671.

Hoskins, B. J., & Karoly, D. J. (1981). The steady linear response of a spherical atmosphere to thermal and orographic forcing. Journal of the Atmospheric Sciences, 38(6), 1179-1196.

Karoly, D. J. (1983). Rossby wave propagation in a barotropic atmosphere. Dynamics of Atmospheres and Oceans, 7(2), 111-125.

Yang, G. Y., & Hoskins, B. J. (1996). Propagation of Rossby waves of nonzero frequency. Journal of the atmospheric sciences, 53(16), 2365-2378.

`ray_path` *Calculate the ray paths / segment of great circles*

Description

This function calculates the segments great circles using the (lat, lon) coordinates obtained with `ray` or `ray_source`. It returns a LINESTRING geometry that is ready for plot.

Usage

```
ray_path(x, y)
```

Arguments

`x`	vector with the longitude obtained with `ray` or `ray_source`
`y`	vector with the latitude obtained with `ray` or `ray_source`

Value

sfc_LINESTRING sfc

Examples

```
{
# Coelho et al. (2015):
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
b <- betaks(u = input)
rt <- ray(betam = b$betam,
          u = b$u,
          lat = b$lat,
          K = 3,
          itime = 30,
          x0 = -135,
          y0 = -30,
          dt = 6,
          direction = -1)
rp <- ray_path(x = rt$lon, y = rt$lat)
plot(rp,  axes = TRUE, graticule = TRUE)
}
```

ray_source *Calculate the Rossby waves ray paths over a source region*

Description

ray_source returns the Rossby wave ray paths (lat/lon) triggered from one or more initial source/position (x0, y0), one or more total wavenumber (K), and one or more direction set up when invoking the function. ray_source must ingest the meridional gradient of the absolute vorticity in mercator coordinates**betam**, the zonal mean wind **u**, and the latitude vector (**lat**). Those variables can be obtained (recommended) using betaks function. The zonal means of the basic state will be calculated along the **ray** program, as well as the conversion to mercator coordinates of **u**. The resultant output is a spatial feature object from a combination of initial and final positions/sources, total wavenumbers (K), and directions.

Usage

```
ray_source(
  betam,
  u,
  lat,
  x0,
  y0,
  K,
  cx,
  dt,
  itime,
  direction,
  interpolation = "trin",
  tl = 1,
  a = 6371000,
  verbose = FALSE,
  ofile
)
```

Arguments

`betam` matrix (longitude = rows x latitude from minor to major = columns) obtained with `betaks`. **betam** is the meridional gradient of the absolute vorticity in mercator coordinates

`u` matrix (longitude = rows x latitude from minor to major = columns) obtained with `betaks`. Is the zonal wind speed in the appropriate format for the `ray`. It will be converted in mercator coordinates inside the `ray`

`lat` Numeric vector of latitudes from minor to major (ex: -90 to 90). Obtained with `betaks`

`x0` Vector with the initial longitudes (choose between -180 to 180)

`y0` Vector with the initial latitudes

`K` Vector; Total Rossby wavenumber

`cx` numeric. Indicates the zonal phase speed. The program is designed for eastward propagation (cx > 0) and stationary waves (cx = 0, the default).

`dt` Numeric value; Timestep for integration (hours)

`itime` Numeric value; total integration time. For instance, 10 days times 4 times per day

`direction` Vector with two possibilities: 1 or -1 It controls the wave displacement: If 1, the wave goes to the north of the source; If -1, the wave goes to the south of the source.

`interpolation` Character. Set the interpolation method to be used: `trin` or `ypos`

`tl` Numeric value; Turning latitude. Do not change this! It will always start with a positive tl (1) and automatically change to negative (-1) after the turning latitude.

`a` Earth's radio (m)

`verbose` Boolean; if TRUE (default) return messages during compilation

`ofile` Character; Output file name with .csv extension, for instance, "/user/ray.csv"

Value

sf data.frame

Examples

```
## Not run:
#do not run
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
b <- betaks(u = input)
rt <- ray_source(betam = b$betam,
                 u = b$u,
                 lat = b$lat,
                 K = 3,
                 itime = 10*4,
                 cx = 0,
                 x0 = -c(130, 135),
                 y0 = -30,
                 dt = 6,
                 direction = -1,
                 interpolation = "trin")

# Plot:
data(coastlines)
plot(coastlines,
     reset = FALSE,
     axes = TRUE,
     graticule = TRUE,
     col = "grey",
     main = "Coelho et al. (2015): JFM/2014")
plot(rt[sf::st_is(rt, "LINESTRING"),]["lon_ini"],
     add = TRUE,
     lwd = 2,
     pal = colorRampPalette(c("black", "blue")))

## End(Not run)
```

`trin` *Performs trigonometric interpolation*

Description

This function performs trigonometric interpolation for the passed basic state variable and the requested latitude

Usage

```
trin(y, yk, mercator = FALSE)
```

Arguments

`y`	Numeric. The latitude where the interpolation is required

`yk` Numeric vector of the data to be interpolated. For instance, umz or betam

`mercator` Logical. Is it require to transform the final data in mercator coordinates? Default is FALSE.

Value

Numeric value

Note

This function is an alternative to `ypos` and is more accurate

See Also

`ypos` `ray` `ray_source`

Other Interpolation: `ypos()`

Examples

```
{
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
b <- betaks(u = input)
umz <- rev(colMeans(b$u, na.rm = TRUE))*cos(rev(b$lat)*pi/180)
betamz <- rev(colMeans(b$betam, na.rm = TRUE))
y0 <- -17
trin(y = y0, yk = umz)
}
```

`wave_arrival` *Filter the ray paths that arrives in an area of interest*

Description

`wave_arrival` ingests the ray paths to filter by determined area of interest. Default CRS 4326.

Usage

```
wave_arrival(x, aoi = NULL, xmin, xmax, ymin, ymax, ofile)
```

Arguments

`x` sf data.frame object with the LINESTRINGS to be filtered.

`aoi` String giving the path and the filename of the area of interest. By default is NULL. If no **aoi** is not provided, the xmin, xmax, ymin, and ymax must be provided.

`xmin` Numeric. Indicates the western longitude to be used in the range -180 to 180.

`xmax`	Numeric. Indicates the eastern longitude to be used in the range -180 to 180.
`ymin`	Numeric. Indicates the southern longitude to be used in the range -90 to 90.
`ymax`	Numeric. Indicates the northern longitude to be used in the range -90 to 90.
`ofile`	Character; Output file name with .csv extension, for instance, "/user/aoi_ray.csv"

Value

sf data.frame

Examples

```
{
}
```

---

ypos *Interpolation selecting the nearest neighbor*

Description

This function get the position in a vector of a given latitute y.

Usage

```
ypos(y, lat, yk, mercator = FALSE)
```

Arguments

`y`	numeric value of one latitude
`lat`	numeric vector of latitudes from minor to major
`yk`	numeric vector to be approximated
`mercator`	Logical. Is it require to transform the final data in mercator coordinates? Default is FALSE.

Value

The position where the latitude y has the minor difference with lat

See Also

Other Interpolation: `trin()`

Examples

```
{
input <- system.file("extdata",
                     "uwnd.mon.mean_200hPa_2014JFM.nc",
                      package = "raytracing")
b <- betaks(u = input)
ykk <- rev(colMeans(b$betam))
ypos(y = -30, lat = seq(90, -90, -2.5), yk = ykk)

}
```

Index