

Package ‘cyclones’

February 14, 2012

Version 1.2-13

Date January 04 2012

Title Cyclone identification

Author Rasmus E. Benestad <rasmus.benestad@physics.org>

Maintainer Rasmus E. Benestad <rasmus.benestad@physics.org>

Depends clim.pact, R (>= 2.10)

Suggests ncdf, akima

Description Functions for locating local minima/maxima

License GPL (>= 2)

URL <http://www.r-project.org> <http://cran.r-project.org>

ZipData no

Repository CRAN

Date/Publication 2012-01-04 13:38:24

R topics documented:

Benestad2005	2
CCI	2
cyclstat	5
dX	6
etopo60	9
filters	9
FrontStats	10
Regression-gradients	12
Storms.ERA40	14
withIn	15
Index	17

Benestad2005

Analysis of solar cycle length (SCL).

Description

The R-code used for making the analysis and figures in Benestad, R.E. (2005) 'A review of the solar cycle length estimates' GRL 32 L15714, doi:10.1029/2005GL023621, August 13. Note. the 10.7 cm flux used to be fetched from URL ftp://ftp.ngdc.noaa.gov/STP/SOLAR_DATA/SOLAR_RADIO/FLUX/DAILYPLT.OBS which no longer seems to work.

Usage

```
f10.7cm(url="ftp://ftp.ngdc.noaa.gov/STP/SOLAR_DATA/SOLAR_RADIO/FLUX/Penticton_Adjusted/daily/DAILYPLT.OBS")
sunspots(url="http://sidc.oma.be/DATA/monthssn.dat",plot=TRUE)
Benestad2005()
data(gcr)
```

Arguments

url	URL
plot	Option to produce graphics
na.rm	TRUE: Ignore NAs

Value

A table or lists containing the relevant data.

Author(s)

R.E. Benestad

Examples

```
Benestad2005()
```

CCI

Calculus Cyclone identification.

Description

Identifies cyclones (low pressure systems) in a gridded data set using a Calculus Cyclone Identification (CCI) method (EMS04-A-00146, Benestad, R.E.; Sorteberg, A.; Chen, D. 'Storm statistics derived using a calculus-based cyclone identification method', http://www.cosis.net/members/meetings/sessions/oral_programme.php?p_id=110&s_id=1845, European Meteorological Society AC2, Nice, Sept 28, 2004). Storms are identified with longitude, latitude, and date. Also returned are estimates of local minimum pressure, max pressure gradient near storm, max geostrophic windspeed near storm, and radius of the storm. The storm location is by means of finding where first derivatives of north-south and east-west gradients both are zero. The storm radius is estimated from the points of inflexion along the latitude and longitude lines running through the centre of the storm.

This code is the basis for the analysis carried out by Benestad et al. 'The use of a Calculus-based Cyclone Identification method for generating storm statistics' submitted to Climate Dynamics (2004).

If a north-south or east-west psl (sea level pressure) profile can be approximated as

$$p(\theta) = p_0 + \sum_{i=1}^{N_\theta} [a_\theta(i) \cos(\omega_\theta(i)\theta) + b_\theta(i) \sin(\omega_\theta(i)\theta)]$$

Then the psl gradient can be estimated as:

$$\frac{\partial \hat{p}(\theta)}{\partial \theta} = \sum_{i=1}^{N_\theta} \omega_\theta(i) [-\hat{a}_\theta(i) \sin(\omega_\theta(i)\theta) + \hat{b}_\theta(i) \cos(\omega_\theta(i)\theta)]$$

The gradient in x and y directions are found after the transform

$$\frac{d\hat{p}(x)}{dx} = \frac{1}{a \cos(\phi)} \frac{d\hat{p}(\theta)}{d\theta}$$

and

$$\frac{d\hat{p}(y)}{dy} = \frac{1}{a} \frac{d\hat{p}(\phi)}{d\phi}$$

(Gill, 1982).

stopCCI signals the CCI process to stop in a safe manner by removing the file '.CCI.run' in the run directory.

gradient.wind estimated the gradient wind of the data produced by CCI after Fleagle and Businger (1980) p. 163. (eq 4.27).

Reference: Benestad & Chen (2006) 'The use of a Calculus-based Cyclone Identification method for generating storm statistics', Tellus A, in press. Benestad (2005)

Usage

```
CCI(maxhar=14,lplot=TRUE,nsim=10,fname="data/cyclones.Rdata",
    fielddata="data/nmc_slp.nc",vname="slp",cyclones=TRUE,force365.25=FALSE,
    x.rng=c(-80,40),y.rng=c(5,75),tslice=3652,rad=5,dx=1,dy=1,
    times=NULL,label=NULL,rho=1.293,nc.read.method="retrieve.nc",
    graph.dir="CCI.graphs/",plot.interval=50,EPS=TRUE,verbose=TRUE,accuracy=NULL)
stopCCI()
gradient.wind(storms=NULL,icyclone=1)
```

Arguments

maxhar	number of harmonics used for fit to profile (Fourier truncation)
lplot	TRUE: produce plots.
nsim	Number of simultaneous cyclones identified and saved ordered according to depth/strength.
fname	Name of file containing the stats of cyclones (output).
fielddata	Name of file (netCDF) of gridded data (input)
vname	Variable name of the field in 'fielddata'.
cyclones	TRUE: identifies cyclones, FALSE anticyclones.
force365.25	TRUE - To ensure a 365.25-day year, FALSE evaluates to find the best match: 365.25-day or 360-day (model) year.
x.rng	Longitude region of interest.
y.rng	Latitude region of interest.
tslice	Number of maps read per chunk of data (time slice).
rad	Max radius of cyclone.
dx	Resolution in longitude (in degrees). NULL skips interpolation - uses original grid
dy	Resolution in latitudes (in degrees). NULL skips interpolation - uses original grid
times	Specify time slices for reading chunks of data.
label	Label for ID-purposes.
rho	Density of air for calculation of geostrophic wind.
nc.read.method	Method to read netCDF or other gridded data. Can be set to a custom made routine. The output should be a 'field'-type object.
graph.dir	Name of the directory in which to store the graphical output. Useful for checking the results, e.g. using <code>animate -delay 50 -pause 5 cci*.eps</code> under Linux.
plot.interval	Interval between the generation of each graphical output
EPS	TRUE for generation of encapsulated PostScript graphics, FALSE for PNG bitmaps. PNG bitmaps are useful for animatio (e.g. <code>animate cci*.png</code> ; http://www.imagemagick.org), whereas postscript versions are good for hard copies.
storms	Data object returned by CCI with cyclone statistics.
icyclone	Decides which cyclone to do the gradient wind estimate for (eg 1 is for the deepest cyclone, 2 for the second deepest, etc).
verbose	Print out diagnostics.
accuracy	Not yet finished.

Value

A list: `list(lon=lon, lat, tim, psl, yy, mm, dd, i, label, max.dpsl, max.speed, radius, rad.max.dpsl, dx, dy)`. The subobjects `lon` (longitude: units degrees), `lat` (latitude: units: degrees), `psl` (local minimum pressure: units hPa), `max.dpsl` (pressure gradient: units Pa/m), `max.speed` (windspeed: units m/s if the units of SLP are in hPa), and `radius` (radius of the storm: units km) are `[1:nt, 1:i.sim]`-matrices. `v.grad` is the estimated gradient wind.

Author(s)

R.E. Benestad

Examples

```
## Not run:
# Shell script for running in batch (background process)
#! /bin/bash
cat > paper20e.R << EOF
library(clim.pact)
library(cyclones)
source("cyclones/R/cyclones.R")

a<- Sys.info()
dir <- switch(substr(as.character(a[4]),1,9),
               "saragasso"="/data1/hirham/",
               "stratonim"="/data1/hirham/",
               "virvelvin"="/home/rasmusd/data/data1/")

filename.1 <- "/home/rasmusd/data/ERA40/era40_slp.nc"
fname.1 <- "data/cyclones_ERA40.Rdata"
vname.1 <- "msl"

print("=====< ERA 15 >=====")
CCI(fname="data/cyclones_ERA40_r1.Rdata",dx=1,dy=1,fielddata=filename.1,
     vname=vname.1,label="ERA40: slp 1-degree res.",force365.25=TRUE,lplot=FALSE)
EOF

nice R CMD BATCH --no-save paper20e.R paper20e.out

## End(Not run)
```

cyclstat

*Cyclone statistics***Description**

Produce storm statistics/analysis. This function was used to produce the figures in the paper Benestad, Sorteberg and Chen, 'The use of a Calculus-based Cyclone Identification method for generating storm statistics' accepted by Tellus A (2006). The function produces four plots: cyclstat.eps (a map showing frequency geographic distribution), cyclstat2.eps (time series showing evolution of number), cyclstat3.eps (annual cycle), and cyclstat4.eps (compare with Poisson distribution).

Usage

```
cyclstat(fname=NULL,psl0=1000,topo="etopo60.Rdata",cyclone=TRUE,
         x.rng=c(5,35),y.rng=c(55,72),cmp=FALSE,mon=c(12,1,2),ERA40=TRUE)
windstat(fname=NULL,ws0=20,topo="etopo60.Rdata",cyclone=TRUE,
         x.rng=c(5,35),y.rng=c(55,72),cmp=FALSE,mon=c(12,1,2),ERA40=TRUE)
discard.bad(results)
```

Arguments

fname	Filename of cyclone stats data.
psl0	Threshold value: PSL lower than.
ws0	Threshold value: wind speed exceedance.
topo	Name of file containing topology; if doesn't exist, use data from the cyclone package.
cyclone	Threshold: TRUE -> less than, FALSE -> greater than
x.rng	Interval of longitudes of interest (deg E)
y.rng	Interval of latitudes of interest (deg N)
cmp	Flag set for comparing two storm data sets: one given a fname and the default in cyclones ('data(storms)')
mon	Months to plot a histogram for.
ERA40	True, use 6hr ERA40 as reference, otherwise use 12-hr NMC as reference.
results	CCI-object.

Value

Figures saved in EPS-format: cyclstat.eps, cyclstat2.eps, cyclstat3.eps, and cyclstat4.eps

Author(s)

R.E. Benestad

Examples

```
## Not run:
library(cyclones)
load("data/cyclonesERA40.Rdata")
results.era40 <- results

europe <- cyclstat(results.era40,cmp=TRUE,x.rng=c(-10,20),y.rng=c(40,55))

## End(Not run)
```

Description

dX calculates the x-derivatives for gridded data in longitude-latitude coordinates. Is based on the equations by Stephenson (1961) p. 162

$$\frac{dA}{dx} = \frac{1}{r \cos(\phi)} * \frac{d}{d\theta}$$

where PHI is the latitude in radians and THETA the longitude.

dY calculates the y-derivatives for gridded data in longitude-latitude coordinates.

$$\frac{dA}{dy} = \frac{1}{r} \frac{d}{d\phi}$$

dT calculates the e.g. time derivatives of a time series.

All functions solve for the gradients by fitting coefficients to a Fourier approximation

$$p(x) = a_0 + \sum_i [a_i \cos(w_i x) + b_i \sin(w_i x)]$$

and dp(x)/dx is

$$\frac{dp(x)}{dx} = \sum [w_i (-a_i \sin(w_i x) + b_i \cos(w_i x))]$$

Also see [derivFFT](#).

Version 1.1-6:

(i) Correction to the wave number in dX and dY - previously wave number was scaled as 1/[number of points] and gave incorrect magnitudes for the derivatives, but now the wave number is 1/distance.

(ii) Included a simple confidence check (chk.conf) to set coefficients to zero when +- 1 sd includes zero. This check can be disabled. Note, it is recommended to use the minimum value for maxhar.

(iii) Included new test routines to test the magnitudes of the gradients.

Usage

```
dX(lon,lat,Z,r=6.378e06,maxhar=NULL,mask.bad=TRUE,plot=FALSE,chk.conf=1,accuracy=NULL)
dY(lon,lat,Z,r=6.378e06,maxhar=NULL,mask.bad=TRUE,plot=FALSE,chk.conf=1,accuracy=NULL)
dT(y,maxhar=NULL,plot=FALSE,chk.conf=1)
test.dX(field=NULL,maxhar=7)
test.dY(field=NULL,maxhar=5)
test.dT(y=NULL,maxhar=15)
```

Arguments

lon	Longitudes
lat	Latitude
Z	Field values of which the gradients are sought
r	mean radius of Earth

maxhar	Number of harmonics to include. If NULL, maxhar is set to the length of the series.
mask.bad	TRUE/FALSE. Used to ignore grid boxes with missing data in the end results - set to NA. Otherwise, interpolated values are used.
y	a time series (vector).
plot	plot data and fit
field	Field to which the test is applied (default: DNMI.slp)
chk.conf	If not NULL, test whether conf. int. (\pm chk.conf * sd) includes zero - if so, set coefficient to zero.
accuracy	To set the accuracy in determining the zero-crossing (unit: degrees N or E).

Value

A list: list(Z=Z,a=a,b=b,c=c,dZ=dZ,Z.fit=Z.fit,lon=lon,lat=lat) of class "map"

Author(s)

R.E. Benestad

Examples

```
# Construct a test series:
t <- seq(-3,3,by=0.1)
nt <- length(t)
y <- 2*rnorm(nt) + 10*sin(t/4) - 7*sin(t/3) + 4*cos(t) - sin(2*t) + 2*cos(3*t) + 0.5*sin(4*t)
plot(y)

dydt <- dT(y)
lines(dydt$y.fit,col="red")
dydt.2 <- dT(y,maxhar=3)
lines(dydt.2$y.fit,col="blue",lty=2)

# First derivative
plot(dydt$dy,type="l")
lines(dydt.2$dy,col="blue",lty=2)

# Second derivative
dy2dt2<-dT(dydt$dy)
dy2dt2.2<-dT(dydt.2$dy)
plot(dy2dt2$dy,type="l")
lines(dy2dt2.2$dy,col="blue",lty=2)
```

etopo60	<i>Topography with 60-minute resolution. This is a low-resolution version of the more recent 2-minute topography: ETOPO5</i>
---------	--

Description

.

Usage

```
data(etopo60)
```

Format

```
list: ETOPO60X, ETOPO60Y, ROSE
```

Source

<http://www.ngdc.noaa.gov/mgg/global/etopo5.HTML> and

References

<http://www.ngdc.noaa.gov/mgg/bathymetry/predicted/explore.HTML>

Examples

```
data(etopo60)
```

filters	<i>Various filters.</i>
---------	-------------------------

Description

A bundle of window types. See e.g. Press et al. (1989), Numerical Recipes in Pascal, pp. 466. i Moving-average (box-car) filter, ii Gaussian filter with cut-off at 0.025 and 0.975 of the area, iii Binomial filter, iv Parzen filter (Press,et al. (1989)), v Hanning filter (Press,et al. (1989)), vi Welch filter (Press,et al. (1989)).

Usage

```
ma.filt(x,n)
gauss.filt(x,n)
binom.filt(x,n)
parzen.filt(x,n)
hanning.filt(x,n)
welch.filt(x,n)
```

Arguments

x series to filter
n window width

Value

A vector

Author(s)

R.E. Benestad

Examples

```
N <- 1000
t <- seq(1,N,by=1) * pi/180
x <- sin(2*t) + 0.3*rnorm(N)
plot(x)
lines(ma.filt(x,15),col="grey40",lwd=4)
lines(gauss.filt(x,15),col="red",lwd=4,lty=2)
lines(binom.filt(x,15),col="blue",lwd=2)
lines(parzen.filt(x,15),col="darkgreen",lwd=2,lty=2)
lines(hanning.filt(x,15),col="darkred")
lines(welch.filt(x,15),col="darkblue",lty=2)
legend(0,-1.2,c("ma.filt(x,15)","gauss.filt(x,15)","binom.filt(x,15)",
"parzen.filt(x,15)","hanning.filt(x,15)","welch.filt(x,15)"),cex=0.7,
lty=c(1,2,1,2,1,2),lwd=c(4,4,2,2,1,1),bg="grey95",
col=c("grey40","red","blue","darkgreen","darkred","darkblue"))
```

FrontStats

Ocean currents and sea surface temperature fronts

Description

The function `FrontStats` uses `dY` to identify the latitudinal position of the sharpest temperature (sea surface temperature=SST) gradient (dT/dy) along a series of north-south transects. Gulfstream and Kuroshio use `FrontStats` to generate a set of coefficients describing the shape and position of the Gulfstream and the Kuroshio ocean currents, based on the position of the sharpest latitudinal SST gradient (dT/dy). The position and the shape of the ocean current - the sharpest latitudinal SST gradient - is described approximately by equation

$$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \beta_5 x^5 + \beta_6 x^6$$

.

The best-fit solution is derived using multiple linear regression (lm).

A demonstration/validation of these results is available as an animation at YouTube: <http://www.youtube.com/watch?v=SQJBa7ynUpw> and <http://www.youtube.com/watch?v=FgDgKhgJ5WA>

Usage

```

FrontStats(sst,plot=FALSE,plot.int=100,maxhar=6,neg.gradient=TRUE)
Gulfstream(fname="sst.wkmean",plot=FALSE,maxhar=12,plot.int = 100)
Kuroshio(fname="sst.wkmean",plot=FALSE,maxhar=12,plot.int = 100)
CurrentClimatology(current="gulfstream",parameter=1)
CurrentHistory(current="gulfstream",parameter=1)
CurrentSpectrum(current="gulfstream",parameter=1,frequency=FALSE)
Current2monthly(current="gulfstream",parameter=1)
CurrentLatitude(current="gulfstream",lon=NULL,nx=50,plot=TRUE)
CurrentValidation(current="gulfstream",fname="sst.wkmean",plot=TRUE,las=1)
data(gulfstream)
data(kuroshio)

```

Arguments

sst	A field object (here the sea surface temperature; SST)
plot	Flag for plotting
plot.int	Plotting interval
maxhar	Number of harmonics to include. See dX .
neg.gradient	Look for negative gradients (decreasing SST with latitude)
fname	Name pattern of netCDF files with SST.
lon	a range of longitude coordinates defining reion of interest
nx	length of
las	as las in par
current	Name of current system ("gulfstream"/"kuroshio")
parameter	parameter/coefficient of the polynomial fit to SST front
frequency	plot spectrum against frequency or periodicity.

Value

A matrix containing the 9 regression coefficients, the R-squared value and the 95-percentile of dT/dy.

Author(s)

R.E. Benestad

Examples

```

## Not run:
sst <- retrieve.nc("sst.wkmean.1981-1989.nc",x.rng=c(-80,-30),y.rng=c(30,50),v.nam="sst")
coefs <- FrontStats(sst,plot=plot,maxhar=16,plot.int=1)

Gulfstream() -> gulfstream; save(file="gulfstream.rda",gulfstream)

Kuroshio() -> kuroshio; save(file="kuroshio.Rdata",kuroshio)

```

```
# This call only works completely for Linux platform with ImageMagick and
# gifmerge installed. Type 'which convert' and 'which gifmerge' to see
# if they are installed. These can be obtained from:
# http://www.imagemagick.org/
# URL: http://the-labs.com/GIFMerge/
#
CurrentValidation()

## End(Not run)
```

Regression-gradients *REGression based CALculus for empirical data*

Description

Use an n-order polynomial fit to model a data series. This polynomial then provides the basis for the calculus: derivations and integrations. One way of estimating slopes in the terrain or the geostrophic winds components from sea level pressure fields.

coefFit fits the coefficients of a power series

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots$$

, where

$$x = -1, \dots, +1$$

. Reference: R.E. Benestad (2003) What can present climate models tell us about climate change? Climatic Change Vol 59, 311-332

coefDeriv computes the first derivative of a power series

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots$$

using the results from coefFit so that

$$dy/dx = a_1 + 2a_2x + 3a_3x^2 + \dots$$

. (Reference:e.g. G. Stephenson (1961), Mathematical methods for science students, Logman Scientific & Technical, p. 86).

coefInt integrates a series

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots$$

using the results from coefFit so that

$$\int ydx = a_0x + 1/2a_1x^2 + 1/3a_2x^3 + 1/4a_3x^4 + \dots + const$$

geoGrad computes gradients of the topography. The topography data file can be obtained from <http://ferret.pmel.noaa.gov/NVODS/servlets/dataset>.

testReg.cal is a test-function for coefFit, coefDeriv and coefInt.

derivFFT uses FFT to compute the first derivative. Analogous to `dx`

integrFFT uses FFT to integrate a series and is the (pseudo) inverse of derivFFT.

testFFTcalc is a test function for derivFFT and integrFFT.

Usage

```
coefFit(y,x=NULL,n=length(y),method="lm")
coefDeriv(y)
coefInt(y,c1=0)
geoGrad(maxhar = 35, fname = "data/etopo5_scandinavia.Rdata",
        x.rng = c(-10, 35), y.rng = c(50, 73), plot = FALSE)
testReg.cal(i.y=240,n=50)
derivFFT(y)
integrFFT(y)
testFFTcalc()
```

Arguments

y	A vector.
x	A vector of index values (e.g. time or longitude).
n	Number of harmonics to fit
method	Regression model for fitting the harmonics
c1	first coefficient: the constant term
maxhar	number of harmonix (see dX)
i.y	index for selecting latitude for testing.
x.rng	longitude range
y.rng	latitude range
fname	Name of topology file
plot	Plot the results

Value

Various results. E.g. derivFFT and integrFFT return vectors with class 'complex'; geoGrad lists a list containing information used for gridding; coefFit returns a list containing best fit ('y.hat') as well as some statistics.

Author(s)

R.E. Benestad

Examples

```
## Not run:
# Polynomial series calculus. (testReg.cal)
library(clim.pact)
load("data/etopo5_scandinavia.Rdata")
y <- ROSE[i.y,]
a <- coefFit(y,n=N)
da <- coefDeriv(a)
a.2 <- coefInt(da,c1=a$coefs[1])
newFig()
```

```

plot(ETOP05X,y,type="s",lwd=3,xlab="Longitude (degE)",ylab="m.a.s.l.",
     main=paste("Transect: ",round(ETOP05Y[i.y],1),"degE"))
polygon(c(ETOP05X,ETOP05X[1320],ETOP05X[1]),
        c(da$y.deriv/quantile(da$y.deriv,0.9)*quantile(y,0.7),0,0),col="blue")
lines(ETOP05X,y,type="s",lwd=4)
lines(ETOP05X,a$y.hat,col="red",lty=2,lwd=2)
lines(ETOP05X,a.2$y.int,col="steelblue",lty=1)

# FFT-based calculus (testFFTcalc)

x <- seq(-3,3,length=100)
y <- 3*cos(5*x) + 0.3*sin(18*x) - 1.4*sin(3*x) + 0.15*sin(23*x)
plot(x,y,type="l",lwd=3)
grid()
dydx <- derivFFT(y)
y2 <- integrFFT(dydx)
lines(x,y2,col="red",lty=2,lwd=2)
grid()
newFig()
plot(x,Im(dydx),type="l",lwd=3)
grid()
lines(x,rep(0,length(x)),col="grey")

## End(Not run)

```

Storms.ERA40

North Atlantic cyclone statistics.

Description

Counts of low-pressure systems every 12hr over the North Atlantic region (00.00 and 12.00 UTC), derived from the NMC (now National Centers for Environmental Prediction, NCEP, <http://www.ncep.noaa.gov/>) or ECMWF re-analysis ERA40 6-hr or 24-hr slp fields ('x1' is once a day).

Usage

```

data(Storms.ERA40)
data(Storms.NMC)

```

Format

'Storms.ERA40' - a list of lon (longitude in degrees), lat (latitude in degrees), psl (central sea level pressure in hPa), yy (year), mm (month), dd (day), max.dpsl (max pressure gradient), tim (time index), max.speed (estimated maximum geostrophic speed), radius (radius of low pressure system in km), rad.max.dpsl, dx, dy, version, scale.factor.x, scale.factor.y. 'storms' - a list of lon, lat, psl, yy, mm, dd. See [CCI](#).

Source

The Norwegian Meteorological Institute, Climatology division.

References

EMS04-A-00146, Benestad, R.E.; Sorteberg, A.; Chen, D. 'Storm statistics derived using a calculus-based cyclone identification method', http://www.cosis.net/members/meetings/sessions/oral_programme.php?p_id=110&s_id=1845, European Meteorological Society AC2, Nice, Sept 28, 2004.

Examples

```
data(Storms.NMC)
summary(Storms.NMC)
cyclstat(Storms.NMC)
```

withIn	<i>Points within an area.</i>
--------	-------------------------------

Description

Finds points within a given closed 2-D boundary.

Usage

```
withIn(x,y,X,Y,test=FALSE,plot=FALSE)
testWithIn(a=3,b=5)
```

Arguments

x	vector with x-coordinates to test if they lie inside given boundary.
y	vector with y-coordinates to test...
X	vector of x-coordinates of the boundary.
Y	vector of y-coordinates of the boundary.
plot	TRUE if plotting results.
test	TRUE if conducting test.
a	x-axis of ellipse.
b	y-axis of ellipse.

Value

Boolean vector of same length as x.

Author(s)

R.E. Benestad

Examples

```
x <- rnorm(100)
y <- rnorm(100)
X <- cos(2*pi*seq(0,1,length=360))
Y <- cos(2*pi*seq(0,1,length=360))
print(sum(withIn(x,y,X,Y)))
```

Index

*Topic **datasets**

etopo60, 9
Storms.ERA40, 14

*Topic **manip**

Benestad2005, 2
CCI, 2
cyclstat, 5
dX, 6
filters, 9
FrontStats, 10
Regression-gradients, 12
withIn, 15

Benestad2005, 2

binom.filt (filters), 9

CCI, 2, 14

coefDeriv (Regression-gradients), 12

coefFit (Regression-gradients), 12

coefInt (Regression-gradients), 12

Current2monthly (FrontStats), 10

CurrentClimatology (FrontStats), 10

CurrentHistory (FrontStats), 10

CurrentLatitude (FrontStats), 10

CurrentSpectrum (FrontStats), 10

CurrentValidation (FrontStats), 10

cyclstat, 5

derivFFT, 7

derivFFT (Regression-gradients), 12

discard.bad (cyclstat), 5

dT (dX), 6

dX, 6, 11–13

dY (dX), 6

etopo60, 9

f10.7cm (Benestad2005), 2

filters, 9

FrontStats, 10

gauss.filt (filters), 9

gcr (Benestad2005), 2

geoGrad (Regression-gradients), 12

gradient.wind (CCI), 2

Gulfstream (FrontStats), 10

gulfstream (FrontStats), 10

hanning.filt (filters), 9

integrFFT (Regression-gradients), 12

Kuroshio (FrontStats), 10

kuroshio (FrontStats), 10

ma.filt (filters), 9

par, 11

parzen.filt (filters), 9

Regression-gradients, 12

stopCCI (CCI), 2

Storms.ERA40, 14

Storms.NMC (Storms.ERA40), 14

sunspots (Benestad2005), 2

test.dT (dX), 6

test.dX (dX), 6

test.dY (dX), 6

testFFTcalc (Regression-gradients), 12

testReg.cal (Regression-gradients), 12

testWithIn (withIn), 15

welch.filt (filters), 9

windstat (cyclstat), 5

withIn, 15