Package 'AOI'

August 24, 2018

Type Package

Title Areas of Interest

Version 0.1.9000

Maintainer Mike Johnson <jmj00@ucsb.edu>

BugReports https://github.com/mikejohnson51/AOI/issues

Description An area of interest (AOI) is a geographic extent.
AOIs help confine and formalize a unit of work to a geographic area and define research and sub setting efforts while improving reproducibility.
They are built around concrete spatial attributes but often are discussed in a more colloquial way.
This package lets users define regions through a common query to achieve spatial geometries.
Tools are provided to help define, describe, and convert points, boundaries, and features to usable forms including strings and political boundaries (for USA states and counties).
In addition, this package provides geocoding and reverse geocoding functions through the Google Maps, OpenStreetMaps and ESRI Webservices.
This package is provided to support the bredth of spatial packages in the R ecosystem.

Depends R(>= 3.3.0),
leaflet

Imports jsonlite,
magrittr,
sf(>= 0.6-0),
utils,
xml2

Suggests testthat

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

URL https://github.com/mikejohnson51/AOI/

R topics documented:

`AOI` *AOI Package*

Description

An area of interest (AOI) is a geographic extent and the aim of this package is to help users create these. The package is written using the simple features paradigm, however, by default, objects are returned as `SpatialPolygons` projected to EPSG:4269. For those that have made the jump to sf, all functions include a 'sf' parameter that can be set to TRUE and eventully the default behavior will change.

The primary functions to be aware of are `geocode`, `getAOI`, `getBoundingBox`. The first returns a set spatial points, the second a single spatial geometry, and the last a geometry encompassing all input features. `bbox_st` and `bbox_sp` help convert AOIs between string and geometry manifestations; `check` helps users visualize AOIs in a interactive leaflet map; and `buffer` allows for the modification of AOIs by uniform distances. Finally, `revgeocode` provides a reverse geocoding interface to help understand coordinates as locations, and `describe` breaks existing spatial features into getAOI parameters to improve the reproducibility of geometry generation.

Two core datasets are served with the package reflecting the spatial geometries of US `states` and `counties`.

See the README on github, and a webpage of examples here.

`aoiProj` *AOI Projection*

Description

Base projection used for all AOI calls: *EPSG:4269*. 'aoiProj = "+init=epsg:4269"'

Usage

```
aoiProj
```

Format

An object of class `character` of length 1.

Author(s)

Mike Johnson

---

bbox_sp *Convert bounding box string to geometry*

Description

Convert a vector, dataframe, bb, or raster object to a spatial geometry

Usage

```
bbox_sp(bbox_st, sf = FALSE)
```

Arguments

bbox_st	a comma seperated character string, numeric vector, or data.frame in the order ("xmin","xmax", "ymin", "ymax"). Raster objects are also accepted.
sf	logical. If TRUE object returned is of class sf, default is FALSE and returns SpatialPolygons

Value

a bounding box geometry

Author(s)

Mike Johnson

Examples

```
## Not run:

## SpatialPolygon from string
   bbox = bbox_sp("37,36,-119,-118")

## SpatialPolygon from vector
   bbox = c(37,38,-119,-118) %>% bbox_sp()

## Simple Feature Polygon from data.frame
   bbox = data.frame(xmin = 37, xmax = 38, ymin = -119, ymax = -118) %>% bbox_sp(sf = T)

## SpatialPolygon from Reverse Geocoding results
   bbox = revgeocode("Santa Barbara")$bb %>% bbox_sp()

## String to Geometry to String (full circle)
    bbox = c(37,38,-119,-118) %>% bbox_sp() %>% bbox_st()

## Raster to sf
   raster %>% bbox_sp(sf = TRUE)

## End(Not run)
```

bbox_st *Convert spatial geometry to string (data.frame)*

Description

Convert a spatial object to a data.frame of (xmin, xmax, ymin, ymax)

Usage

```
bbox_st(AOI)
```

Arguments

AOI	any spatial object (raster, sf, sp). Can be piped (%>%) from getAOI

Value

a bounding box data.frame (xmin. xmax, ymin, ymax)

Author(s)

Mike Johnson

Examples

```
## Not run:
## Get a bounding box data.frame for AOI
   AOI = getAOI(list("UCSB", 10, 10)) %>% bbox_st()

## End(Not run)
```

buff *Buffer AOI*

Description

Add or subtract a uniform distance to/from a spatial obeject in either miles or kilometers.

Usage

```
buff(AOI, d, km = FALSE)
```

Arguments

AOI	a spatial, raster or simple features object
d	numeric.The distance by which to modify each edge
km	logical.Is the distance in kilometers? Default is FALSE and in miles

Value

a spatial geometry of the same class as the input AOI (if Raster sp returned)

Author(s)

Mike Johnson

Examples

```
## Not run:
 # get an AOI of 'Garden of the Gods' and add a 2 mile buffer
    getAOI("Garden of the Gods") %>% buffer(10)

 # get an AOI of 'Garden of the Gods' and add a 2 kilometer buffer
    getAOI("Garden of the Gods") %>% buffer(10, km = TRUE)

 # get and AOI for Colorado Springs and subtract 3 miles
    getAOI("Garden of the Gods") %>% buffer(-3)

## End(Not run)
```

---

check *Generate Leafet map and tool set*

Description

Generate an interactive `leaflet` map for checking, and refining AOI queries. Useful `leaflet` tools allow for the marking of points, measuring of distances, and interactive panning and zooming to help define an approapriate AOI.

Usage

```
check(AOI = NULL)
```

Arguments

AOI	any spatial object (`raster`, `sf`, `sp`). Can be piped (%>%) from `getAOI`

Value

a `leaflet` html object

Author(s)

Mike Johnson

Examples

```
## Not run:
## Generate an empty map:
     check()

## Check a defined AOI:
     AOI = getAOI(clip = list("UCSB", 10, 10))
     check(AOI)
```

```
## Chain to AOI calls:
     getAOI(clip = list("UCSB", 10, 10)) %>% check()

## Add layers with standard leaflet functions:
     r = getAOI("UCSB") %>%  # get AOI
     HydroData::findNED() %>%  # get raster of elevation data
     HydroData::findNWIS() # get SpatialPointsDataframe of local USGS gages

     check(r$NED) %>% addMarkers(data = r$nwis, popup = r$nwis$site_no)

## Save map for reference:
     m = getAOI("Kansas City") %>% check()
     htmlwidgets::saveWidget(m, file = paste0(getwd(), "/myMap.html"))

## End(Not run)
```

`counties` *USA Counites*

Description

Dataset containing `SpatialPolygons` of USA Counties. Data is initalized from the USAboundaries and USAboundariesData package, converted to `SpatialPolygons`, re=projected and cleaned-up for this package. The primary reason for doing this to provide a more minimalistic dataset primed for this package and leaflet use.

Usage

```
counties
```

Format

a `SpatialPolygonsDataFrame`, 3220 observations of 7 variables

- 'statefp': A `character` State 2-digit Federal Information Processing Standards (FIPS) code
- 'countyfp': A `character` County 3-digit Federal Information Processing Standards (FIPS) code
- 'affgeoid': A `character` AFF Summary Level Code
- 'geoid': A `character` Concatinates state and county FIP code
- 'name': A `character` County name
- 'state_name': A `character` State name
- 'state_abbr': A `character` State Abbriviation

Source

USAboundaries

Examples

```
## Not run:
 AOI::counties

## End(Not run)
```

describe *Describe an AOI*

Description

Describes a spatial, raster or sf object in terms of a reproducable clip area.

Usage

```
describe(AOI)
```

Arguments

AOI	any spatial object (raster, sf, sp). Can be piped (%>%) from getAOI

Value

a data.frame of AOI descriptors including:

latCent the AOI center latitude

lngCent the AOI center longitude

height height in (miles)

width width in(miles)

origin AOI origin

name Most descriptive place name from revgeocode

Author(s)

Mike Johnson

Examples

```
## Not run:
 AOI = getAOI(clip = list("UCSB", 10, 10)) %>% describe()

 ```
 latCent : 34.4139629
 lngCent : -119.848947
 height  : 10 miles
 width   : 10 miles
 origin  : center
 name    : 93106, Santa Barbara, California
 ```

End(Not run)
```

geocode *Geocoding*

Description

A wrapper around the Google and OpenSteetMap geocoding web-services. Users can request a lat/long pair, spatial points, and/or a bounding box geometry.

One or more locations can be given at a time. If a single point is requested, 'geocode' will provide a matix of lat lon vals; a spatial point and the geocode derived bounding box (if requested). If multiple points are given the returned objects will be a matrix with columns for input name-lat-lon; a SpatialPoints object; and a minimum bounding box of input locations.

Usage

```
geocode(location = NULL, pt = FALSE, bb = FALSE, sf = FALSE,
  server = "google")
```

Arguments

location	character. Place name(s)
pt	logical. If TRUE points geometery is appended to the returned list()
bb	logical. If TRUE bounding box geometry is appended to the returned list()
sf	logical. If TRUE object(s) returned are of class sf, default is FALSE and returns sp
server	character. What server should be prioritized. Options inlcude 'google' or 'osm' (default = 'google')

Value

at minimum a matrix of lat/long coordinates. Possible list with appended spatial features of type sf or sp

Author(s)

Mike Johnson

Examples

```
## Not run:
 ## geocode a single location
     geocode("UCSB")
 ## geocode a single location and return a SpatialPoints object
     geocode("UCSB", pt = TRUE)
 ## geocode a single location and derived bounding box of location
     geocode("UCSB", bb = TRUE)
 ## geocode multiple locations
     geocode(c("UCSB", "Goleta", "Sterns Warf"))
 ## geocode multiple points and generate a minimum bounding box of all locations
     geocode(c("UCSB", "Goleta", "Sterns Warf"), bb = T, pt= T)

## End(Not run)
```

getAOI *Get Area of Interest (AOI) geometry*

Description

Generate a spatial geometry from:

1. US state name(s)
2. US state, county pair(s)
3. a user spatial, sf or raster object
4. a clip unit (see details)

Usage

```
getAOI(clip = NULL, state = NULL, county = NULL, sf = FALSE,
  km = FALSE, bb = FALSE)
```

Arguments

clip	A spatial, raster, sf or a list object (see details for list parameters)
state	character. Full name or two character abbriviation. Not case senstive
county	character. County name(s). Requires state input. Not case senstive. If 'all' then all counties in a state are returned
sf	logical. Ilogical. If TRUE object returned is of class sf, default is FALSE and returns SpatialPolygons
km	logical. If TRUE distances are in kilometers, default is FALSE with distances in miles
bb	logical. Only applicable for state and county calls. If TRUE the bounding geometry of state/county is returned, default is FALSE and returns fiat geometries

Details

A clip unit can be described by just a location (eg 'UCSB'). In doing so the associated boundaries determined by geocode will be returned. To have greater control over the clip unit it can be defined as a list with a minimum of 3 inputs:

1. A point:
   - 'location name' (character) ex: "UCSB"
   - 'lat/lon' pair: ex: "c(-36, -120)"
2. A bounding box height (numeric)
   - in miles ex: 10
3. A bounding box width (numeric)
   - in miles ex: 10

The bounding box is always drawn in relation to the location. By default the point is treated as the center of the box. To define the realtive location of the point to the bounding box, a fourth input can be used:

1. Origin
    - 'center' (default)
    - 'upperleft'
    - 'upperright'
    - 'lowerleft'
    - 'lowerright'

In total, 1 to 5 elements can be used to define `clip` element and **ORDER MATTERS** (point, height, width, origin). Acceptable variations include:

- 1 member: (1) location name
    - *"UCSB"*
- 3 members: (1) location name, (2) height, (3) width
    - *list("UCSB", 10, 10)*
- 4 members: (1) latitude, (2) longitude, (3) height, (4) width
    - *list(36, -120, 10, 10)*
- 4 members: (1) location name, (2) height, (3) width, (4) origin
    - *list("UCSB", 10, 10, "lowerright)*
- 5 members: (1) lat, (2) long, (3) height, (4) width, (5) origin
    - *list(36,-120, 10, 10, "upperright)*

Value

a geometry projected to *EPSG:4269*.

Author(s)

Mike Johnson

Examples

```
## Not run:
# Get AOI for a location
    getAOI("Sacramento")

# Get AOI defined by a state(s)
    getAOI(state = 'CA')
    getAOI(state = c('CA', 'nevada'))

# Get AOI defined by state & county pair(s)
    getAOI(state = 'California', county = 'Santa Barbara')
    getAOI(state = 'CA', county = c('Santa Barbara', 'ventura'))

# Get AOI defined by state & county pair(s)
    getAOI(state = 'California', county = 'Santa Barbara')
    getAOI(state = 'CA', county = c('Santa Barbara', 'ventura'))

# Get AOI defined by external spatial file:
    getAOI(clip = sf::read_sf('la_metro.shp'))
    getAOI(clip = raster('AOI.tif'))

# Get AOI defined by 10 mile bounding box using lat/lon
```

```
    getAOI(clip = c(35, -119, 10, 10))

# Get AOI defined by 10 mile2 bounding box using the 'KMART near UCSB' as lower left corner
    getAOI(clip = list('KMART near UCSB', 10, 10, 'lowerleft'))

## End(Not run)
```

`getBoundingBox` *Get mimimum bounding box of spatial features*

Description

Returns a minimum bounding box for a spatial, raster or sf object(s)

Usage

```
getBoundingBox(x, sf = FALSE)
```

Arguments

`x`	a `data.frame` with a lat and long column, a raster, sf, or spatial object
`sf`	`logical`. If TRUE object returned is of class `sf`, default is FALSE and returns `SpatialPolygons` Default is `FALSE` and returns class SpatialPolygon

Author(s)

Mike Johnson

Examples

```
## Not run:
  ## Find the 10 closest Airports to UCSB
     ap = geocode("UCSB") %>% HydroData::findNearestAirports(n =10)
     AOI = ap$ap %>% getBoundingBox()

  ## Get bounding box of raster object
     AOI = getBoundingBox(r)

## End(Not run)
```

`revgeocode` *Reverse Geocoding*

Description

Describe a location using the ERSI and OSM reverse geocoding web-services. This service provides tradional reverse geocoding (lat/long to placename) but can also be use to get more information about a place name.

Usage

```
revgeocode(point)
```

Arguments

point	a point provided by numeric lat,long pair or character place name

Value

a data.frame of descriptive features

Author(s)

Mike Johnson

Examples

```
## Not run:
 revgeocode(c(38,-115))

 ```
 county          : Lincoln Count
 state           : Nevada
 country         : United States of America
 place_id        : 198776170
 osm_type        : relation
 osm_id          : 166463
 lat             : 37.5449476
 lon             : -114.8764448
 bb              : -115.897545,-114.048473,36.8420756,38.678486
 match_addr      : 89017, Hiko, Nevada
 longlabel       : 89017, Hiko, NV, USA
 shortlabel      : 89017
 addr_type       : Postal
 city            : Hiko
 countrycode     : USA
 ```

 revgeocode("UCSB")

 ````
 university      : UCSB
 pedestrian      : Library Plaza
 county          : Santa Barbara County
 state           : California
 postcode        : 93106
 country         : United States of America
 place_id        : 187839690
 osm_type        : way
 osm_id          : 542863702
 lat             : 34.4145937
 lon             : -119.84581949869
 bb              : -119.8851155,-119.8360437,34.4047282,34.4243918
 match_addr      : 93106, Santa Barbara, California
 longlabel       : 93106, Santa Barbara, CA, USA
 addr_type       : Postal
 city            : Santa Barbara
 countrycode     : USA
 ````
```

```
## End(Not run)
```

`states` *USA States*

Description

Dataset containing `SpatialPolygons` of USA States. Data is initalized from the USAboundaries and USAboundariesData package, converted to `SpatialPolygons`, re-projected and cleaned-up for this package. The primary reason for doing this is to provide a more minimalistic dataset primed for this package and leaflet use.

Usage

```
states
```

Format

a `SpatialPolygonsDataFrame`, 52 observations of 5 variables

- 'statefp': A `character` State 2-digit Federal Information Processing Standards (FIPS) code
- 'statens': A `character` American National Standards Institute (ANSI) code
- 'affgeoid': A `character` AFF Summary Level Code
- 'state_name': A `character` State Name
- 'state_abbr': A `character` State Abbriviation

Source

USAboundaries

Examples

```
## Not run:
AOI::states

## End(Not run)
```

Index