Package 'rasterbc'

May 9, 2026

Type Package

Title Access Forest Ecology Layers for British Columbia in 2001-2018

Version 1.0.2

Description R-based access to a large set of data variables relevant to forest ecology in British Columbia (BC), Canada. Layers are in raster format at 100m resolution in the BC Albers projection, hosted at the Federated Research Data Repository (FRDR) with <doi:10.20383/101.0283>. The collection includes: elevation; biogeoclimatic zone; wildfire; cutblocks; forest attributes from Hansen et al. (2013) <doi:10.1139/cjfr-2013-0401> and Beaudoin et al. (2017) <doi:10.1139/cjfr-2017-0184>; and rasterized Forest Insect and Disease Survey (FIDS) maps for a number of insect pest species, all covering the period 2001-2018. Users supply a polygon or point location in the province of BC, and 'rasterbc' will download the overlapping raster tiles hosted at FRDR, merging them as needed and returning the result in R as a 'SpatRaster' object. Metadata associated with these layers, and code for downloading them from their original sources can be found in the 'github' repository <https://github.com/deankoch/rasterbc_src>.

License MIT + file LICENSE

URL https://github.com/deankoch/rasterbc, https://github.com/deankoch/rasterbc_src

BugReports https://github.com/deankoch/rasterbc/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports sf, methods, terra

Depends R (>= 2.10)

Suggests knitr, rmarkdown, bcmaps, units

VignetteBuilder knitr

NeedsCompilation no

Author Dean Koch [aut, cre] (ORCID: <https://orcid.org/0000-0002-8849-859X>)

Maintainer Dean Koch <dkoch@ualberta.ca>

Repository CRAN

Date/Publication 2023-11-09 05:50:02 UTC

Contents

`datadir_bc` *Set local directory for storage of raster data files*

Description

All files downloaded/created by the rasterbc package are written to the directory assigned by this function. The path to this directory is stored in the global options list for R under 'rasterbc.data.dir'. To set this path, call `datadir_bc` and pass the path as argument `data.dir`. To check the current setting, call `datadir_bc()` without arguments.

Usage

```
datadir_bc(data.dir = NULL, quiet = FALSE)
```

Arguments

`data.dir`	character string, the absolute path to the desired storage directory
`quiet`	logical indicating to skip confirmation prompt and warnings about existing directories

Details

If `data.dir=NA`, the data storage directory is set to the a subdirectory of `base::tempdir()`, a per-session temporary directory (cleared after each R session). This happens automatically if `opendata_bc` or `getdata_bc` are called before `datadir_bc`, ensuring that `rasterbc` won't unexpectedly overwrite things or leave garbage in the user's file system.

However, users are strongly encouraged to set the data directory manually to a non-temporary location. This allows copies of downloaded data to persist between sessions, so that `rasterbc::opendata_bc`

can load the local copies in future R sessions. This is both much faster than downloading mapsheets repeatedly, and it reduces the strain on FRDR's data hosting service.

The directory `data.dir` will be created if it doesn't already exist.

Value

character string, the absolute path to the data storage directory

Examples

```
# call without arguments to get the current directory (or a prompt to set it, if unset)
datadir_bc()
# call with argument NA to set a default location
datadir_bc(NA, quiet=TRUE)
datadir_bc()
```

`findblocks_bc` *Identify NTS/SNRC codes covering a given geometry*

Description

Data layers for the province of BC are split according to the mapsheet coding system used by Canada's NTS/SNRC. This function identifies mapsheets intersecting with the geographical extent of `geo`.

Usage

```
findblocks_bc(geo = NULL, type = "character")
```

Arguments

`geo`	A point, line, or polygon object of class `sfc`, or a character vector of 4-character codes
`type`	character, the object type to return, either 'character' or 'sfc'

Details

When `type='character'`, the function returns the 4-character NTS/SNRC codes corresponding to mapsheets that intersect with the input geometry. When `type='sfc'`, the function returns the mapsheet polygons themselves.

`geo=NULL` (the default) indicates to return all mapsheet codes/polygons. Non-NULL `geo` must either be a character vector of codes, or else an sfc class object having a coordinate reference system ('crs' attribute) defined. If `geo` intersects with none of the BC mapsheets, (or contains unknown NTS/SNRC codes) the function returns `NULL`.

Value

either a character vector or an sfc object (see details)

Examples

```
# list all mapsheet codes then print the corresponding sfc object
findblocks_bc()
findblocks_bc(type='sfc')

# define an example point by specifying latitude and longitude (in WGS84 reference system)
input.point = sf::st_point(c(x=-120, y=50)) |> sf::st_sfc(crs='EPSG:4326')

# this point lies at the intersection of four mapsheets, which are in Albers projection
blocks = findblocks_bc(input.point, type='sfc')
blocks |> sf::st_geometry() |> plot()
sf::st_transform(input.point, crs='EPSG:3005') |> plot(add=TRUE)
blocks |> sf::st_geometry() |> sf::st_centroid() |>
    sf::st_coordinates() |> text(labels=blocks$NTS_SNRC)

# nudge the point slightly so it intersects with only one mapsheet
input.point = sf::st_point(c(x=-120.1, y=50.1)) |> sf::st_sfc(crs='EPSG:4326')
blocks |> sf::st_geometry() |> plot()
sf::st_transform(input.point, crs='EPSG:3005') |> plot(add=TRUE)
blocks |> sf::st_geometry() |> sf::st_centroid() |>
   sf::st_coordinates() |> text(labels=blocks$NTS_SNRC)
findblocks_bc(input.point)

# make a polygon (circle) from the point and repeat
if( requireNamespace('units', quietly = TRUE) ) {
input.polygon = input.point |> sf::st_buffer(units::set_units(10, km))
blocks |> sf::st_geometry() |> plot()
sf::st_transform(input.polygon, crs='EPSG:3005') |> plot(add=TRUE)
blocks |> sf::st_geometry() |> sf::st_centroid() |>
   sf::st_coordinates() |> text(labels=blocks$NTS_SNRC)
findblocks_bc(input.polygon)
}

# geo can be a character vector of codes
input.codes = c('093A', '093I', '104O')
findblocks_bc(input.codes, type='sfc')
```

`getdata_bc` *Download data from the rasterbc collection*

Description

Downloads all mapsheet layers (geoTIFFs) covering the geographical extent of `geo` for the specified `collection`, `varname`, and `year`. Input `geo` can be a vector of (4-character) NTS/SNRC mapsheet codes or a geometry of class `sfc`.

Usage

```
getdata_bc(
  geo = NULL,
  collection = NULL,
  varname = NULL,
  year = NULL,
  force.dl = FALSE,
  quiet = FALSE
)
```

Arguments

geo	vector of character strings (NTS/SNRC codes) or a geometry of class sfc
collection	character string indicating the data collection to query
varname	character string indicating the layer to query
year	integer indicating the year to query (if applicable)
force.dl	logical indicating whether to overwrite any existing data
quiet	logical, suppresses console messages

Details

The data files are written to the directory returned by `rasterbc::datadir_bc()`. Mapsheets found there (already downloaded) are not downloaded again unless `force.dl==TRUE`. Users should only need to download a mapsheet once - there are no plans to update the rasterbc collection in the future.

Value

a vector of character string(s) containing the absolute path(s) to the downloaded file(s)

See Also

findblocks_bc to identify which mapsheets will be downloaded

listdata_bc for a list of available collections, variable names, years

Examples

```
# define a location of interest, and a polygon around it then fetch the corresponding DEM data
input.point = sf::st_point(c(x=-120.1, y=50.1)) |> sf::st_sfc(crs='EPSG:4326')

if( requireNamespace('units', quietly = TRUE) ) {
input.polygon = input.point |> sf::st_buffer(units::set_units(10, km))

## Not run:
# the following downloads data from FRDR
block.path = getdata_bc(input.point, 'dem')
getdata_bc(input.polygon, 'dem')

# load one of the mapsheets
```

```
terra::rast(block.path)

## End(Not run)
}
```

`listdata_bc` *List all available layers, or their associated filepaths and download status*

Description

Returns a list of dataframes (one per collection) describing the variables available through rasterbc, or a subset (as specified by `collection`, `varname`, `year`). Alternatively, if `simple==TRUE`, returns a nested list indicating the filepaths associated with these layers, and which of them exist in the local data storage directory already.

Usage

```
listdata_bc(
  collection = NULL,
  varname = NULL,
  year = NULL,
  verbose = 1,
  simple = FALSE
)
```

Arguments

`collection`	(Optional) character string, indicating the data collection to query
`varname`	(Optional) character string, indicating the layer to query (see Details, below)
`year`	(Optional) integer or character string, indicating the year to query (see Details, below)
`verbose`	An integer (0, 1, 2), indicating how much information about the files to print to the console
`simple`	logical indicating to return a (list of) logical vector(s) indicating existence on disk of specific filenames

Details

The layers available through this package are organized into "collections", corresponding to their original online sources. Layers in a collection are further organized by variable name, and are uniquely identified by the character string `varname` (and, if applicable, `year`). The optional arguments `collection`, `varname`, `year` prompt this function to return only the applicable subsets.

Value

Either a (list of) dataframe(s) containing information about each raster layer, or (when `simple==TRUE`) a nested list of logical values (named according to filepath), with entries for each data file in the specified subset.

Examples

```
# print available collections
listdata_bc() |> names()

# print info about a specific collection
listdata_bc('bgcz')
listdata_bc('bgcz', verbose=2)

# example with a year field
listdata_bc('fids', varname='IBM_mid', year=2005, verbose=2)
listdata_bc('fids', varname='IBM_mid', verbose=2)

# "simple=TRUE" mode returns logical vector indicating which mapsheets are downloaded
listdata_bc(collection='dem', varname='aspect', verbose=2, simple=TRUE)
```

---

metadata_bc *Metadata for rasterbc collections*

Description

This file contains a summary of metadata about the available datasets in rasterbc. The script ('metadata_bc.R') used to generate this file can be found in the subdirectory rasterbc/data-raw/ (last updated November 09, 2023).

Usage

```
metadata_bc
```

Format

Nested list containing metadata and URLs for all of the files available as rasterbc collections

bgcz BC biogeoclimatic zone from BC Ministry of Forests

borders Geographical coordinates grid from Natural Resources Canada

cutblocks Consolidated cutblocks, 2001-2018, from BC Ministry of Forests

dem Digital elevation model from Natural Resources Canada

fids Forest insect and disease survey, 2001-2018, from BC Ministry of Forests

gfc Forest extent and change, 2001-2019 from Hansen et al. (2013)

nfdb Canadian national fire database, 2001-2018, from Natural Resources Canada

pine Interpolated forest attributes, 2001, 2011, from Beaudoin et al. (2017) (DOI: 10.1139/cjfr-2017-0184)

Details

Relevant contents of the file can be accessed dataframe using the function listdata_bc.

The rasterbc collection is published as a data publication with associated DOI for permanence and easy referencing. For a more complete description, along with instructions on downloading the collections from their sources and reproducing the collection, see the rasterbc_src github repository.

All were downloaded from sources and processed in the years 2018-2020, then stored as raster tiles in the standard BC Albers projection, and hosted on FRDR.

Source

Original data sources were published by the Canadian Journal of Forest Research and various Canadian government environment ministries, and are described in full at rasterbc_src

---

`ntspoly_bc` *National Topographic System Index Maps for British Columbia.*

Description

Mapsheet boundary polygons for 1:250,000 scale maps, from Natural Resources Canada. The Open Government License - Canada applies. More info available here

Usage

```
ntspoly_bc
```

Format

Simple feature collection (of type POLYGON), with 89 features and 1 field:

NTS_SNRC four-character mapsheet code

Source

Reproduced (in NAD83 / BC Albers projection) from the shapefile 'nts_snrc_250k.shp' in the zip archive 'nts_snrc.zip' available from http://geogratis.gc.ca/ (accessed June 11, 2020).

opendata_bc *Load/merge data blocks and optionally clip/mask them*

Description

Loads all mapsheets covering the geographical extent of input argument `geo`. This can be a vector of (4-character) NTS/SNRC block codes, or a geometry of class `sfc` having a defined coordinate reference system.

Usage

```
opendata_bc(
  geo = NULL,
  collection = NULL,
  varname = NULL,
  year = NULL,
  type = "mask",
  quiet = FALSE,
  dl = TRUE
)
```

Arguments

`geo`	vector of character strings (NTS/SNRC codes) or a geometry of class `sfc`
`collection`	character string, indicating the data collection to query
`varname`	character string, indicating the layer to query
`year`	integer, indicating the year to query
`type`	character string, one of 'all', 'clip', 'mask'
`quiet`	logical, suppresses console messages
`dl`	logical, enables automatic downloading of missing files

Details

Data for the layer specified by `collection`, `varname`, and (as needed) `year`, are fetched from the directory specified by `datadir_bc`, merged into a single (mosaic) layer, cropped and masked as needed, and then loaded into memory and returned as a `SpatRaster` object. If the files are not found, and `dl=TRUE`, they will be automatically downloaded.

When `geo` is a line or point type geometry (or when `type='all'`), the function uses `terra::merge` to create a larger (mosaic) SpatRaster containing the data from all mapsheets intersecting with the input extent.

When `geo` is a polygon, `type` can be set to clip or mask the returned raster: 'all' returns the mosaic, as above; 'clip' crops the mosaic and to the bounding box of `geo`; and 'mask' (the default) crops the mosaic then sets all points not lying inside `geo` to NA. Note that `type` is ignored when `geo` is a point geometry or a character string of codes (these cases behave like `type='all'`).

Value

A SpatRaster

Examples

```
# define a location of interest, and a circle of radius 10km around it
input.point = sf::st_point(c(x=-120.1, y=50.1)) |> sf::st_sfc(crs='EPSG:4326')

if( requireNamespace('units', quietly = TRUE) ) {
input.polygon = input.point |> sf::st_buffer(units::set_units(10, km))

## Not run:
# the following downloads data from FRDR
# open the DEM mapsheets corresponding to the polygon and plot
opendata_bc(geo=input.polygon, 'dem') |> terra::plot()

## End(Not run)
}
```

Index