RAP Data download and focal mean extraction

Download and processing of Rangeland Analysis Platform rasters

Currently configured for version 2. May be able to change "v2" to "v3" in RAP_Path but haven't tested it yet.

RAP_Extraction.R

Extracts data from RAP based on extents of shp file

CRS has to be lat/long for gdalwarp Specific to Oswalt but can be modified by loading a different extents shape file. This file should be a rectangle/square polygon. Uses maxx, maxy, minx, miny coordinates of extents file.

Be aware that you may have to tinker with the CRS some to get gdalwarp to work with a different shape file.

Includes options to download Vegetative cover, Biomass, and Woody transitions. Woody transitions rasters are available at two resolutions - 30m and 240m

Writes rasters to various folders depending on data set
Plots and writes pdfs to same folders

RAP_Focal_Means_v2.R - updated to do multiple focal filters

Focal/Moving window spatial analysis of RAP data for NOBO project Calculates Focal mean for RAP bands

Source tif files need to have 4 digit year embedded in filename - don't change year in file names from RAP_Extraction.R The NeonScience site is a pretty useful resource here.

Writes to "NOBO_Focal_Means_df.dat" and "NOBO_Focal_Means_df.csv"

- Source Data required:
  - shape files
    - polygons for masking and buffering

    - point for extraction

  - Rasters - should work with any of the RAP layers regardless of extent - as long as they have same extent

Number of cells in focal window - depends on resolution and size of window needed
Example: 17X17 - so 17 cells * 30m resolution == 510m on a side

Matrix() references these values to determine size of filter. Currently set to 3 different resolutions.

```
ScaleList <- c(3, 17, 35) # Set number of cells for focal mean filter

w = matrix(1, nrow = z, ncol = z) "z" in loop references values in ScaleList
```

CRS is set to EPSG: 4326

```
`crs_str <- 4326  ##  This solves the proj4 update issues`
```

Workflow:

RAP_Extraction.R

Load shape file for deriving extents. If gps data is involved, locations are probably the file to load - they are likely to occur outside of any property line/boundary and so will have a greater extent. All you need is a rectangle that encompasses your area of interest.

To change the file for extents, modify lines ~40-51 for your shape file. Load, transform, buffer if necessary.

```
Oswalt <- st_read(paste0(source_path, "/boundary.shp"))

# Buffer before reprojecting
Oswalt_1500       <- st_buffer(Oswalt, dist = 1500) # 1500m buffer on Oswalt boundary

Oswalt_1500_WGS84 <- st_transform(Oswalt_1500, crs_str)  ##  Reproject vectors
Oswalt_WGS84 <- st_transform(Oswalt, crs_str)  ##  Reproject vectors

# These are used for plotting - change to match your vectors
boundary_shape <- Oswalt_WGS84      # original boundary
buffer_shape   <- Oswalt_1500_WGS84 # buffered boundary
```

Getting the extents right for the gdalwarp function was the biggest issue with this script. CRS causes problems here as well - gdalwarp wants lat/long.

Using st_bbox() instead of extent() solved the te=aoi issue with coords being out of order.

Downloading multiple data sets will require multiple passes through parts of this script.

Run these designated sections of code to select time frame and data set. The loop will download rasters based on the extents from the shape file provided, the range of years, and the selected data set.

To download additional data sets for same time period just run the related code block for that dataset and rerun the loop.

Select years to download - modify Years if necessary. This is set separately for woody transition layers

```
RAP_year <- c(2019)      ## a single year
RAP_year <- c(2007:2019) ## a range of years
```

Vegetation cover - This is the original cover data set with 6 bands

```
RAP_path    <- "http://rangeland.ntsg.umt.edu/data/rap/rap-vegetation-cover/v2/vegetation-cover-v2-"
RAP_path
raster_path <- file.path(base_path, "source_data/veg_cover//")
raster_type <- "Cover"
```

Biomass

```
RAP_path    <- "http://rangeland.ntsg.umt.edu/data/rap/rap-vegetation-biomass/v2/vegetation-biomass-v2-
RAP_path
raster_path <- file.path(base_path, "source_data/biomass//")      # Writing to these folders now
raster_type <- "Biomass"
```

Woody transitions - there are two resolutions available. A separate pass is required for each.

240m resolution

```
RAP_path    <- "http://rangeland.ntsg.umt.edu/data/rap/rap-derivatives/great-plains/woody-transitions/t
RAP_path
raster_type <- "Woody_trans_240"
raster_path <- file.path(base_path, "source_data/woody_trans/240//")
RAP_year <- c(1990, 2000, 2010, 2015:2019)## This is all years available for woody transitions
```

30m resolution

```
RAP_path    <- "http://rangeland.ntsg.umt.edu/data/rap/rap-derivatives/great-plains/woody-transitions/t
RAP_path
raster_type <- "Woody_trans_30"
raster_path <- file.path(base_path, "source_data/woody_trans/30//")
RAP_year <- c(1990, 2000, 2010, 2015:2019)## This is all years available for woody transitions
```

Two additional loops provide a means of plotting the downloaded rasters and generating pdf files of them. It is recommended to run both immediately after each data set. Otherwise you'll have to go back and set the correct parameters again.

Although the files from all data sets are uniquely named so that they "could" be put in the same folder they're not. Each data set has its own folder. If downstream processing involves loading entire folders of files at once, all of those files need to have the same number of bands. Keeping the cover rasters (which have 6 bands) separate facilitates this. The other data sets could be combined into one folder without issue (I think)

RAP_Focal_Means_v2.R

Converts rasters to data frames of focal mean values. This file is used as a base to add other topographic, edaphic, or anthropomorphic layers and calculate focal means for all layers. There are a lot of things that happen a lot of times. Most of the loops might have been better as functions but due to technical or organizational issues were left as loops. Cells for focal window can be changed - see file description above. Updates to Proj library caused much grief - solution seems to be to use EPSG codes for any crs transformations.

Source tif files need to have 4 digit year embedded in filename

1. Creates new rasterstack by adding shrub and tree layers.

2. Calculates Focal Mean for each layer and Extracts Focal Mean values based on point shape file.

- Outline of file
  - Load rasters
  - Set projection to utm
  - Convert lists to RasterStacks
  - Add shrub and tree layers together in new RasterStack
  - Read in shapes
  - Mask to buffer
  - Stack of all layers for plotting etc
  - DF of woody layer to check against orig_values

  – Calculate Focal Means for each layer

  – Generate plots of focal mean

  – Generate pdfs of above

  – Extract values based on points from focal mean rasters
  – Write data files

  – Plotting

  – Barplot of focal mean across years

  – Animations

While not needed in this process, *GetExtentsFromRasterList.R* has code that could be useful for downstream analysis. If the extents of all layers don't line up you won't be able to make a raster stack. This makes using functions to extract data difficult.

GetExtentsFromRasterList.R will load rasters from a folder, loop through them, plot and and print extents for each one. Examining the output will indicate discrepancies in extents. Either fix the extents or process the odd ones separately. No guidance on fixing extents - some I managed to fix, others I had to process individually.

```
# Generates package citations
library(grateful)
library(tidyverse)
library(lubridate)
library(magrittr)
library(rprojroot)
library("tidylog", warn.conflicts = FALSE)
library(raster)
library(sf)
library(stringr)
library(tictoc)
library(animation)

cite_packages()

#  There is an entry in the YAML as well that makes this work
```

References

Bache, Stefan Milton, and Hadley Wickham. 2020. *Magrittr: A Forward-Pipe Operator for r.* https://CRAN.R-project.org/package=magrittr.

Elbers, Benjamin. 2020. *Tidylog: Logging for 'Dplyr' and 'Tidyr' Functions.* https://CRAN.R-project.org/package=tidylog.

Greenberg, Jonathan Asher, and Matteo Mattiuzzi. 2020. *gdalUtils: Wrappers for the Geospatial Data Abstraction Library (GDAL) Utilities.* https://CRAN.R-project.org/package=gdalUtils.

Grolemund, Garrett, and Hadley Wickham. 2011. "Dates and Times Made Easy with lubridate." *Journal of Statistical Software* 40 (3): 1–25. https://www.jstatsoft.org/v40/i03/.

Hijmans, Robert J. 2021. *Raster: Geographic Data Analysis and Modeling*. https://CRAN.R-project.org/package=raster.

Izrailev, Sergei. 2021. *Tictoc: Functions for Timing r Scripts, as Well as Implementations of Stack and List Structures*. https://CRAN.R-project.org/package=tictoc.

Müller, Kirill. 2020. *Rprojroot: Finding Files in Project Subdirectories*. https://CRAN.R-project.org/package=rprojroot.

Pebesma, Edzer. 2018. "Simple Features for R: Standardized Support for Spatial Vector Data." *The R Journal* 10 (1): 439–46. https://doi.org/10.32614/RJ-2018-009.

R Core Team. 2021. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Rodríguez-Sánchez, Francisco, and Shaurita D. Hutchins. 2020. *Grateful: Facilitate Citation of r Packages*. https://github.com/Pakillo/grateful.

Wickham, Hadley. 2019. *Stringr: Simple, Consistent Wrappers for Common String Operations*. https://CRAN.R-project.org/package=stringr.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2021. *Dplyr: A Grammar of Data Manipulation*. https://CRAN.R-project.org/package=dplyr.

Wickham, Hadley, and Jim Hester. 2021. *Readr: Read Rectangular Text Data*. https://CRAN.R-project.org/package=readr.

Xie, Yihui. 2014. "Knitr: A Comprehensive Tool for Reproducible Research in R." In *Implementing Reproducible Computational Research*, edited by Victoria Stodden, Friedrich Leisch, and Roger D. Peng. Chapman; Hall/CRC. http://www.crcpress.com/product/isbn/9781466561595.

Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. *R Markdown Cookbook*. Boca Raton, Florida: Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown-cookbook.

Xie, Yihui, Christian Mueller, Lijia Yu, and Weicheng Zhu. 2021. *Animation: A Gallery of Animations in Statistics and Utilities to Create Animations*. https://yihui.org/animation/.