ECOGRAPHY

lsatTS: an R package for generating vegetation greenness time series using Landsat satellite data

Journal:	*Ecography*
Manuscript ID	Draft
Wiley - Manuscript type:	Software Note
Keywords:	NDVI, Google Earth Engine, vegetation greening, vegetation browning, vegetation indices, phenology

Abstract:

The Landsat satellites provide near-global surface reflectance measurements since the early 1980s that are increasingly used to assess interannual changes in terrestrial ecosystem function. These assessments often rely on spectral indices (e.g., NDVI) related to vegetation greenness and productivity. Nevertheless, multiple factors impede multi-decadal assessments of spectral indices using Landsat satellite data, including ease of data access and cleaning, as well as lingering issues with cross-sensor calibration and challenges with irregular timing of cloud-free acquisitions. To help address these problems, we developed the lsatTS package for R. This software package facilitates sample-based time series analysis of surface reflectance and spectral indices derived from Landsat sensors. The package includes functions that enable the extraction of the full Landsat record for point sample locations or small study regions using the Google Earth Engine accessed directly from R. Moreover, the package includes functions for (1) rigorous data cleaning, (2) cross-sensor calibration with machine learning, (3) phenological modeling, and (4) time series analysis. For an example application, we show how lsatTS can be used to assess changes in annual maximum vegetation greenness from 2000 to 2020 across a study area on Disko Island in the Greenlandic Arctic. Overall, this software provides a suite of functions to enable broader use of Landsat satellite data for assessing and monitoring terrestrial ecosystem function over the past four decades across local to global geographic extents.

Note: The following files were submitted by the author for peer review, but cannot be converted to PDF. You must view these files (e.g. movies) online.

disko_1_extract_lsat_data.R
disko_2_anlz_lsat_data.R

Abstract

The Landsat satellites provide near-global surface reflectance measurements since the early 1980s that are increasingly used to assess interannual changes in terrestrial ecosystem function. These assessments often rely on spectral indices (e.g., NDVI) related to vegetation greenness and productivity. Nevertheless, multiple factors impede multi-decadal assessments of spectral indices using Landsat satellite data, including ease of data access and cleaning, as well as lingering issues with cross-sensor calibration and challenges with irregular timing of cloud-free acquisitions. To help address these problems, we developed the *lsatTS* package for R. This software package facilitates sample-based time series analysis of surface reflectance and spectral indices derived from Landsat sensors. The package includes functions that enable the extraction of the full Landsat record for point sample locations or small study regions using the Google Earth Engine accessed directly from R. Moreover, the package includes functions for (1) rigorous data cleaning, (2) cross-sensor calibration with machine learning, (3) phenological modeling, and (4) time series analysis. For an example application, we show how *lsatTS* can be used to assess changes in annual maximum vegetation greenness from 2000 to 2020 across a study area on Disko Island in the Greenlandic Arctic. Overall, this software provides a suite of functions to enable broader use of Landsat satellite data for assessing and monitoring terrestrial ecosystem function over the past four decades across local to global geographic extents.

Background

Ecological monitoring using the Landsat satellites

Satellite remote sensing is crucial for assessing and monitoring how Earth's terrestrial ecosystems have changed during recent decades (National Academies of Sciences 2018). The Landsat satellites are particularly valuable in this regard because they are the longest continuously running satellite program and were designed for terrestrial ecosystem monitoring at moderate spatial resolution (Wulder et al. 2019). The first Landsat satellite (Landsat 1) was launched in 1972 as a partnership between NASA and the US Geological Survey (USGS) and since that time a series of additional satellites have been launched, with the most recent being Landsat 9 in 2021. The Landsat satellites carry multispectral sensors that provide surface reflectance measurements used for a wide range scientific and land management applications (Wulder et al. 2019). These include, for instance, global monitoring of forest canopy cover (Hansen et al. 2013, Sexton et al. 2013) and surface water extent (Pekel et al. 2016), as well as regional- to biome-scale assessments of how land-use and climate change are impacting terrestrial ecosystems (Pastick et al. 2019, Wang and Friedl 2019, Berner et al. 2020, Berner and Goetz 2022). Hence, the Landsat program has become a cornerstone of Earth surface monitoring.

Impediments to Landsat time series analyses

In recent years, it has become easier to access, process, and analyze Landsat data; however, there are still challenges that hinder use of these data by ecologists, land managers, and other non-remote sensing specialists. The USGS made the Landsat archive publicly available in 2008 (Woodcock et al. 2008) and in recent years Google has hosted a copy of the archive accessible via the cloud-computing platform Google Earth Engine (GEE; Gorelick et al. 2017). These steps have made Landsat data much more readily available to the end user and enabled time series analyses of the Normalized Difference Vegetation Index (NDVI) and other spectral indices of "vegetation greenness" that are related to productivity (Tucker 1979, Goetz and Prince 1999, Berner et al. 2020, Camps-Valls et al. 2021). However, time series analyses that use

measurements from multiple sensors are hindered by systematic biases in both individual bands and spectral indices among the Landsat 5 Thematic Mapper (TM), Landsat 7 Enhanced Thematic Mapper Plus (ETM+), and Landsat 8 Operational Land Imager (OLI) sensors (Ju and Masek 2016, Roy et al. 2016, Berner et al. 2020, Berner and Goetz 2022). If unaccounted for, these biases can introduce strong artificial trends into combined time series, such as spurious increases in NDVI over time ("greening") (Sulla-Menashe et al. 2017). Existing approaches for cross-sensor calibration focus on linear corrections (Ju and Masek 2016, Roy et al. 2016), but not all relationships are linear, and corrections are available for a limited number of spectral indices (e.g., NDVI) and often based on regional data. Another potential hindrance when analyzing Landsat time series is the irregular timing of clear-sky acquisitions. This can make it challenging to characterize the NDVI or other spectral indices at a desired phenological stage (e.g., peak summer) and is especially problematic in regions with short growing seasons, such as the rapidly warming Arctic (Berner et al. 2020). Simple calculations of annual maximum NDVI ($NDVI_{max}$) will have a low bias early in the Landsat record, but less so during later years when more observations are available during each growing season. Hence, again, care is needed to avoid the introduction of spurious greening trends into the time series (Berner et al. 2020). In summary, while Landsat data are more readily available than ever before, there are lingering challenges for specialists and non-specialists alike.

The lsatTS package

We developed the R package *lsatTS* (i.e., *Landsat Time Series*) to facilitate sample-based time series analysis of spectral indices derived from surface reflectance measured by multispectral sensors on several Landsat satellites. Specifically, *lsatTS* includes functions for sample-based extraction of full data records from Landsat 5, 7, and 8 that is accomplished by querying the Landsat Collection 2 data set on GEE (Gorelick et al. 2017) using the application programming interface provided by the *rgee* package in R (Aybar et al. 2020). Further functions included in *lsatTS* facilitate (1) data cleaning, (2) cross-sensor calibration with machine learning, (3) characterization of growing season conditions using phenological modeling, and (4) time series analysis of vegetation greenness (Figure 1, Table 1). Altogether, *lsatTS* offers an integrated framework for Landsat data extraction, processing, and time series analysis for sample locations anywhere on Earth's surface.

lsatTS is implemented within the free, open-source, and widely-used R software environment (R Core Team 2021). Several R packages currently exist for processing Landsat data, including *landsat* (Goslee 2011) and *landsat8* (dos Santos 2017). *landsat* includes functions for radiometric and topographic correction of Landsat scenes, while *landsat8* includes functions for computing top of atmosphere reflectance, radiance, and/or brightness temperature on Landsat scenes. These existing packages provide valuable tools for processing individual Landsat scenes. Nevertheless, *lsatTS* provides fundamentally different functionality that includes an integrated framework for robust time-series analysis of vegetation dynamics at local to global scales.

lsatTS grew out of recent research projects that assessed changes in vegetation greenness since the early 1980s for the Arctic tundra and boreal forest biomes (Berner et al. 2020, Berner and Goetz 2022). We found a sample-based approach was well-suited for assessing vegetation dynamics and evaluating ecological hypotheses for these biomes, while substantially reducing computational burden compared with wall-to-wall analyses. Moreover, this sample-based approach enables comparisons between satellite and field measurements across widely distributed site networks (Boyd et al. 2019, Berner et al. 2020, Boyd et al. 2021, Walker et al.

2021) and is conducive to rigorous propagation of uncertainty using Monte Carlo simulations (Berner et al. 2020, Berner and Goetz 2022). Comparisons with field measurements are crucial for validating and interpreting vegetation dynamics inferred from satellites measurements, while uncertainty assessments are crucial for improving confidence in such analyses but are seldom if ever carried out partially because of computational constraints (Myers-Smith et al. 2020). Overall, *lsatTS* provides integrated, sampled-based framework that has recently been used to assess vegetation responses to climate change (Berner et al. 2020, Berner and Goetz 2022), insect outbreaks (Boyd et al. 2019, Boyd et al. 2021), wildfires (Gaglioti et al. 2021), and permafrost degradation (Verdonen et al. 2020) in cold northern biomes.

The following sections detail package installation and summarize the purpose and behavior of each *lsatTS* function. Furthermore, we demonstrate the utility of *lsatTS* with an example application focused on changes in vegetation greenness from 2000 to 2020 across a study area in the Greenlandic Arctic. For a detailed list of function descriptions, including the complete lists of arguments require by each function, please consult the helpfiles provided with the R package or refer to the list of function definitions supplied in the Supplementary Material.

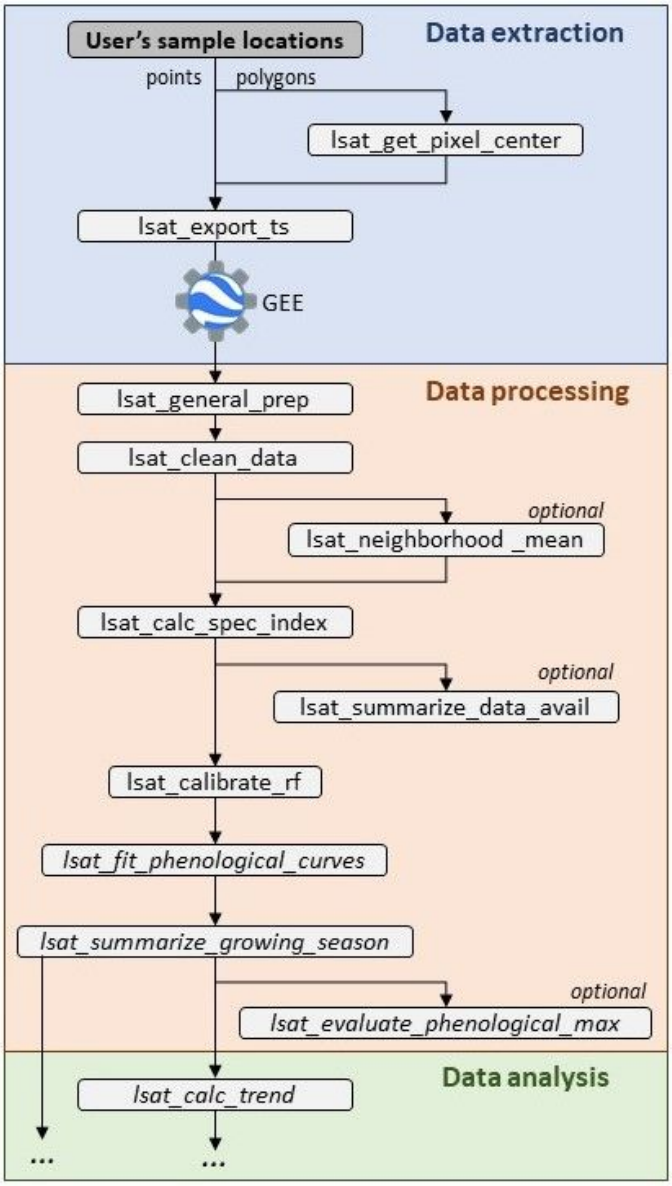

Figure 1. Schematic illustrating functions and typical workflow of the *lsatTS* package. Each function is described in the main text and Table 1. *lsatTS* has primary been used for assessments of interannual variability and trends in vegetation greenness. However, *lsatTS* facilitates other Landsat time series analyses by providing tools for general data extraction and processing.

Table 1. Function names and descriptions. These are listed in the order typically used.

Step	Function	Description
Data extraction	lsat_get_pixel_centers	(*Optional*) Retrieve point coordinates of all Landsat 8 pixel centers that fall within a polygon.
	lsat_export_ts	Export full Landsat surface reflectance time series for a set of point coordinates using GEE accessed from R.
Data processing	lsat_general_prep	Prepare data exported from GEE, including parsing satellite names and renaming and scaling bands.
	lsat_clean_data	Filter out measurements based on presence of clouds, water, shadows, oblique view angles, and other criteria.
	lsat_summarize_data_avail	(*Optional*) Summarize data availability at each site, such as total number and years of observations.
	lsat_neighborhood_mean	(*Optional*) For buffered sites, compute band-wise mean surface reflectance across grid cells within the buffer.
	lsat_calc_spec_index	Calculate a variety of widely used spectral indices, such as the Normalized Difference Vegetation Index (NDVI).
	lsat_calibrate_rf	Cross-calibrate bands or spectral indices from Landsat 5/8 to match Landsat 7 using Random Forests.
	lsat_fit_phenological_curves	Characterize seasonal land surface phenology at each site by iteratively fitting flexible cubic splines.
	lsat_summarize_growing_seasons	Estimate various phenological metrics from fitted cubic splines, such as annual maximum vegetation greenness.
	lsat_evaluate_phenological_max	(*Optional*) Evaluate estimates of annual maximum vegetation greenness with measurement availability.
Data analysis	lsat_calc_trend	Calculate temporal trends using non-parametric Mann-Kendall trend tests and Theil-Sen slope indicators.

Package installation

The R package *lsatTS* is publicly available through a GitHub code repository. Users will need to have installed the R software environment on their computer. The *lsatTS* package is operating system agnostic and can be installed from within R using the *install_github()* function from the *devtools* package:

```
devtools::install_github("logan-berner/lsatTS")
```

The installation will compile the package from source code on the user's computer. As the *lsatTS* package itself is exclusively written in R code, no additional software is required.

To use the data extraction and preparation functions, users will need an account on GEE, and to have installed and configured the *rgee* package to access GEE from R. Please see the GEE (https://earthengine.google.com/) and *rgee* (https://r-spatial.github.io/rgee/) websites for details on signing up for an account and configuring *rgee*, respectively.

All other external package dependencies are configured and automatically dealt with by *devtools* during the installation. These required packages include (*lsatTS* tested with version cited): *magrittr* v2.0.1 (Bache and Wickham 2020), *dplyr* v1.0.7 (Wickham et al. 2021), *tidyr* v1.1.4 (Wickham 2021), *sf* v1.0-4 (Pebesma 2018), *crayon* v1.4.2 (Csárdi 2021), *mapview* v2.10.0 (Appelhans et al. 2021), *purr* v0.3.4 (Henry and Wickham 2020), *data.table* v1.14.2 (Dowle and Srinivasan 2021), *ggplot2* v3.3.5 (Wickham 2016), *R.utils* v2.11.0 (Bengtsson 2021), *stats* v4.1.1 (R Core Team 2021), *stringr* v1.4.0 (Wickham 2019), *ggpubr* v0.4.0 (Kassambara 2020), *ranger*

v0.13.1 (Wright and Ziegler 2017), *zoo* v1.8.9 (Zeileis and Grothendieck 2005), and *zyp* v0.10-1.1 (Bronaugh and Werner 2019).

Data extraction

lsatTS enables point sample-based extraction of full Landsat data records from GEE using the application programming interface provided by the *rgee* package. Sample locations typically represent (1) center coordinates of field sites, (2) a census of all Landsat pixels from a small area of interest, or (3) a random sample from a large region. Data extraction is conducted using the function *lsat_export_ts()*. If the user wishes to extract Landsat data for all pixels in a small area of interest, then the central coordinates of each pixel can be obtained using *lstat_get_pixel_centers()* and then those sample locations are passed to *lsat_export_ts()*. Please note *lsat_export_ts()* has not been tested for data extractions exceeding 10^5 Landsat pixels (~90 km^2). A recent analysis of the boreal forest biome focused on reflectance measurements acquired June through August from 1985 to 2019 for 10^5 Landsat pixels. This data extraction took about two weeks to run on GEE and yielded a total of ~41.6 million multispectral measurements that required ~15 Gb of hard drive storage (Berner and Goetz 2022). *lsatTS* enables large data extractions but is not infinitely scalable.

Export point-coordinate Landsat time series from Google Earth Engine using lsat_export_ts()

The function *lsat_export_ts()* exports time series of Landsat 5, 7 and 8 surface reflectance measurements for each sample location by querying the Landsat Collection 2 archived on GEE. Data are exported for user-defined time periods. It is important to stress this function only works for sample locations (point coordinates) that must be supplied as a simple feature (*sf*) collection of point geometries. The function issues one or more tasks to GEE that export the data in the form of comma separated value (CSV) files to the user's Google Drive. The number of tasks issued varies depending on the number of sample locations for which the Landsat record is to be extracted. Data extractions that involve a large number of sample locations are prone to errors and may exceed user limits set by GEE. Therefore, the function will chunk the sample locations into small groups (by default 250 sites) and for each chunk will issue a separate export task to GEE. The function returns a list of *rgee* task objects, which can be used to query the progress of the exports and subsequently retrieve the data from the user's Google Drive.

Optional: Get central coordinates of pixels within a polygon using lsat_get_pixel_centers()

The function *lsat_get_pixel_centers()* facilitates extracting data for all Landsat pixels in a small area of interest (e.g., < 5 km x 5 km) by determining the central coordinates of all Landsat pixels that fall within a user-specified polygon. The user-specified polygon is supplied to the function as a simple feature collection. The function determines the Landsat Worldwide Reference System (WRS) scene whose center is closest to the center of the user-specified polygon. It then extracts the center coordinates for all pixels that overlap with the user-specified polygon from the first Landsat 8 scene on record available on GEE. A buffer can be specified to include additional pixels beyond the polygon boundary. The function returns the pixel centers as a simple feature object that can then be passed to the *lsat_export_ts()* function for the extraction of the Landsat time series. Please note this function is not designed to be used for sampling polygons that would exceed tens of thousands of Landsat pixels. The number of pixels in large polygons can quickly become too difficult to handle in the subsequent export and processing workflow, and such polygons may also extend beyond the area of the Landsat scene (185 km x 180 km) used to

determine the pixel centers. For large areas, we recommend a random or regular subsampling of point locations such as done in prior studies (Berner et al. 2020, Berner and Goetz 2022).

Data processing

Prepare data for analysis using lsat_general_prep()

The function *lsat_general_prep()* takes the GEE exports generated by *lsat_export_ts()* and prepares the data for the subsequent *lsatTS* workflow. These preprocessing tasks include parsing coordinates and other information, renaming columns, and scaling band values. The GEE exports need to be passed to the function in the form of a *data.table* object. *lsat_general_prep()* returns a *data.table* object that can then be passed on to *lsat_clean_data()* for the next step in the processing workflow. Please note that all *lsatTS* functions handling a *data.table* object require a column called "sample.id" that uniquely identifies each location. If this column is not called "sample.id", please modify accordingly.

Clean surface reflectance data using lsat_clean_data()

The function *lsat_clean_data()* filters measurements to those made under clear-sky conditions. This function allows the user to filter measurements based on pixel quality flags and scene criteria. The USGS provides pixel quality flags based on the CFMask algorithm (Zhu et al. 2015) and information on each scene (e.g., cloud cover). The default settings for *lsat_clean_data()* will filter out measurements flagged as snow or water, as well as measurements acquired at high solar zenith angle (>60°), those with high geolocation uncertainty (>15 m), or those acquired as part of scenes with extensive cloud cover (>80%). Additionally, optional water masking is provided based on maximum surface water extent from the Landsat-based JRC Global Surface Water Dataset (Pekel et al. 2016). The main input supplied to *lsat_clean_data()* is a *data.table* of Landsat records for individual sample locations (specified by a sample.id column) - usually the direct output of lsat_general_prep() - and returns cleaned records in the form of an updated *data.table*, along with a console message summarizing the number and percentage of measurements removed during cleaning (generally >70%).

Compute neighborhood mean surface reflectance using lsat_neighborhood_mean()

The function *lsat_neighborhood_mean()* computes the mean band-specific reflectance across a neighborhood of pixels for measurements at each period in time. This is helpful when each of the user's sample locations were buffered to include a neighborhood of Landsat pixels (e.g., 3 x 3 pixels). The main input to this function is a *data.table* of Landsat records for buffered sample locations. The function returns a new *data.table* with mean reflectance for each band at each point in time at every sample location. If used, the function should be called immediately after *lsat_clean_data()*.

Summarize data availability for each site using lsat_summarize_data_avail()

The function *lsat_summarize_data_avail()* takes a *data.table* of Landsat records and returns a summary *data.table* that provides information on the time period and number of observations available for each sample location. It also generates a figure showing the annual median (2.5^{th} and 97.5^{th} percentile) number of observations available from each satellite summarized across all sample locations. The figure is plotted to the current graphics device and can be saved by calling the function *ggsave()*.

Calculate spectral indices using lsat_calc_spec_index()

The function *lsat_calc_spec_index()* calculates a variety of common spectral indices. The function currently supports calculating 15 spectral indices, including the Normalized Difference Vegetation Index (NDVI), 2-band Enhanced Vegetation Index (EVI2), and others (Table 2). Note the function can only compute one spectral index at a time. As an input it requires a *data.table* with Landsat records and a string indicating the spectral index to be calculated. The function then returns the *data.table* updated with a new column containing the spectral index for each observation.

Table 2. Spectral indices that can be computed using the *lsat_calc_spec_index()* function.

Name	Abbreviation	Formula	Citation
Enhanced Vegetation Index	EVI	$\frac{2.5(NIR - RED)}{NIR + 6 * RED - 7.5 * BLUE + 1}$	Huete et al. (2002)
Enhanced Vegetation Index (2-band)	EVI2	$\frac{2.5 * (NIR - RED)}{NIR + 2.5 * RED + 1}$	Jiang et al. (2008)
Moisture Stress Index	MSI	$\frac{SWIR1}{NIR}$	Rock et al. (1986)
Near Infrared Vegetation Index	NIRv	$\frac{NIR * (NIR - RED)}{NIR + RED}$	Badgley et al. (2017)
Normalized Burn Ratio	NBR	$\frac{NIR - SWIR2}{NIR + SWIR2}$	Key and Benson (1999)
Normalized Difference Infrared Index	NDII	$\frac{NIR - SWIR1}{NIR + SWIR1}$	Hardisky et al. (1983)
Normalized Difference Moisture Index	NDMI	$\frac{NIR - SWIR1}{NIR + SWIR1}$	Gao (1996)
Normalized Difference Vegetation Index (red)	NDVI	$\frac{NIR - RED}{NIR + RED}$	Rouse et al. (1974)
Normalized Difference Vegetation Index (green)	gNDVI	$\frac{NIR - GREEN}{NIR + GREEN}$	Gitelson and Merzlyak (1998)
Normalized Difference Vegetation Index (kernel)	kNDVI	$\tanh\left(\left(\frac{NIR - RED}{NIR + RED}\right)^2\right)$	Camps-Valls et al. (2021)
Normalized Difference Water Index	NDWI	$\frac{GREEN - NIR}{GREEN + NIR}$	McFeeters (1996)
Plant Senescence Reflectance Index	PSRI	$\frac{RED - BLUE}{NIR}$	Merzlyak et al. (1999)
Soil Adjusted Vegetation Index	SAVI	$1.5 * \frac{SWIR1 - RED}{SWIR1 + RED * 0.5} - \frac{SWIR2}{2}$	Huete (1988)
Soil Adjusted Total Vegetation Index	SATVI	$\frac{1.5\ (NIR - RED)}{NIR + RED + 0.5}$	Marsett et al. (2006)
Wide Dynamic Range Vegetation Index	WDRVI	$\frac{NIR - RED}{0.2 * NIR + RED}$	(Gitelson 2004)

Cross-calibrate spectral band or index across sensors using lsat_calibrate_rf()

The function *lsat_calibrate_rf()* will calibrate individual bands or spectral indices from Landsat 5 TM and Landsat 8 ETM+ to match Landsat 7 ETM using random forest models following the approach developed by Berner et al. (2020). Further cross-sensor calibration is needed because there are systematic differences in individual bands and spectral indices among Landsat sensors that must be addressed when combining data from multiple sensors (Ju and Masek 2016, Roy et al. 2016, Berner et al. 2020, Berner and Goetz 2022). Here, the Landsat 7 ETM is used as a benchmark because it temporally overlaps with the other two sensors. Cross-calibration can only be performed on one band or spectral index at a time and requires having data from 100s to

preferably many 1,000s of sample locations to train the random forest models. There is an option for users to train the random forest models using pre-processed Landsat data from ~6000 randomly sampled locations across the Arctic – Boreal domain.

The overall approach involves determining the median spectral reflectance at a sample location during a portion of the growing season using Landsat 7 and Landsat 5/8 data that were collected the same years. A random forest model is then trained to predict Landsat 7 reflectance from Landsat 5/8 reflectance. Random forest models are ensembles of regression trees (Breiman 2001) that here are trained using a fast implementation provided by the *ranger* package (Wright and Ziegler 2017). If the user's dataset includes both Landsat 5 and 8, then the function will train a random forest model for each sensor. The function evaluates model performance using both out-of-bag and cross-validated approaches. Please see Berner et al. (2020) for further details.

The main input to *lsat_calibrate_rf()* is a *data.table* of Landsat records for sample locations and a string specifying the name of the band or spectral index to be cross-calibrated. By default, *lsat_calibrate_rf()* will return a *data.table* with a new column containing the cross-calibrated data. The function creates a user-specified output directory that contains (1) trained random forest models, (2) a CSV file with model evaluation metrics, and (3) a multi-panel figure comparing sensors pre- and post-calibration. Furthermore, model evaluation metrics are returned to the console and the figure plotted in the active graphics device. If the default setting to add a new column with the cross-calibrated data is used, then either use those data in the subsequent functions (e.g., ndvi.xcal) or, once satisfied, manually overwrite the uncalibrated data to simplify subsequent column names.

Fit phenological curves to vegetation greenness time series using lsat_fit_phenological_curves()

The function *lsat_fit_phenological_curves()* provides information on the phenological timing of every Landsat observation relative to multi-year estimates of annual maximum vegetation greenness at each sample location. Specifically, the function models seasonal land surface phenology at each sample location using flexibly cubic splines iteratively fit to vegetation greenness (e.g., NDVI) time series within successive moving-windows. The magnitude and timing of annual maximum vegetation greenness are determined for each time period by first pooling observations over years within each moving-window and then fitting cubic splines to observations that have been sorted by day of year. For each time period, a cubic spline is initially fit that describes vegetation greenness for each day of year during the growing season. To screen outliers, each observation of vegetation greenness is compared against the model fitted values for that day of year and if the deviation is greater than a user-specified difference (default is a 30% difference), then the observation is removed, and the cubic spline is re-fit. This is repeated until no observations exceed the user-specified threshold. The phenological status of each remaining observation is then determined relative to the modeled maximum vegetation greenness during the multi-year period. Additional details are provided in Berner et al. (2020).

The function takes as input a *data.table* with irregular time series of vegetation greenness observations at each sample location, as well as several parameters (e.g., moving window width, minimum number of observation needed to fit a cubic spline, cubic spline flexibility). The function returns a new *data.table* with phenological information for each remaining observation that occurred during a time period with adequate data for modeling surface phenology (i.e., typically fewer observations will be returned than are provided to the function). Among other output, the returned *data.table* provides for each observation the modeled estimates of (1) vegetation greenness for that day of year and for peak summer; (2) vegetation greenness for that

day of year as a fraction of annual maximum vegetation greenness; (3) day of year when annual maximum vegetation greenness occurred; (4) and expected difference in vegetation greenness between that day of year and peak summer. The function also returns a figure to the current graphic device that shows seasonal progression of Landsat observations and modeled surface phenology for a random subset of nine sample locations. The user can optionally output a CSV that includes for each sample location the vegetation greenness predicted for each day of year during each time period by the cubic splines. Furthermore, the function includes an optional "test run" mode that will run the function on a random subset of nine sample locations and return a figure showing model fits, thus allowing the user to quickly experiment with different parameter settings. Note the function was designed to characterize seasonal phenology in terrestrial ecosystems with a single growing season and thus may not be suitable for use in ecosystems with multiple growing seasons. Also, the function was designed for spectral indices that are typically positive (e.g., NDVI). If using a spectral index that is typically negative (e.g., NDWI) then multiply the index by -1 before running the *lsat_fit_phenological_curves()* and *lsat_summarize_growing_seasons()* functions and then back-transform afterwards

Derive annual growing season metrics using lsat_summarize_growing_seasons()

The function *lsat_summarize_growing_seasons()* estimates several annual growing season metrics from vegetation greenness time series and modeled land surface phenology derived from Landsat satellite observations. The function's main input is the *data.table* generated by *lsat_fit_phenological_curves()* and user-specified parameters including the name of the spectral index and the phenological cut-off for an observation to be considered part of the growing season. Specifically, an observation is considered to be part of the growing season if the modeled vegetation greenness for that day of year is within a user-specified fraction of modeled annual maximum vegetation greenness (by default 0.75). The function returns a new *data.table* that includes for each sample location the annual mean, median, and 90th percentile vegetation greenness computed from observations during each growing season. The function also returns phenologically-modeled estimates of the magnitude and timing (day of year) of annual maximum vegetation greenness. For each sample location, annual maximum vegetation greenness is estimated by first adjusting individual observations by the expected difference in vegetation greenness between that day of year and peak summer and then taking the median of phenologically-adjusted values within each growing season. Please see Berner et al. (2020) for additional details.

Assess estimates of maximum vegetation greenness using lsat_evaluate_phenological_max()

The function *lsat_evaluate_phenological_max()* assesses how estimates of annual maximum vegetation greenness vary with the number of Landsat observations when derived from raw observations and after phenological modeling. Raw estimates of annual maximum vegetation greenness are sensitive to the number of observations available from a growing season, but phenological modeling tends to substantially reduce this dependency (Berner et al. 2020). The main input to the function is a *data.table* with Landsat records and phenological information generated by *lsat_fit_phenological_curves()*. The function assumes the "actual" annual maximum vegetation greenness at a sample location is captured by having at least a user-specific number of observations (e.g., ≥ 7). The function extracts site x years with at least the user-specified number of growing season observations and then repeatedly compares how raw and phenologically-modeled estimates of annual maximum vegetation greenness differ from actual

annual maximum vegetation greenness as progressively smaller subsets of observations are used. The function returns a figure to the current graphic device that summarizes how raw and modeled estimates of annual maximum vegetation greenness differ from actual conditions when there are between 1 and n-1 Landsat observations from a single growing season. This lets the user determine how much annual estimates of maximum vegetation greenness are impacted by the number of available growing season observations.

Data analysis

Compute interannual trends in vegetation greenness using lsat_calc_trend()

The function *lsat_calc_trend()* computes a temporal trend in annual time series of vegetation greenness for each sample location over a user-specified time period. This function iteratively pre-whitens each time series (i.e., remove temporal autocorrelation) (Yue et al. 2002) and then computes Mann-Kendall trend tests and Theil-Sen slope indicators as implemented by the *zyp.yuepilon()* function from the *zyp* package (Bronaugh and Werner 2019). The function takes as input a *data.table* with annual time series of vegetation greenness for each sample location. The function returns (1) a new *data.table* that summarizes the interannual trend at each sample location; (2) a console message summarizing trends across all sample locations; and (3) a multi-panel figure summarizing interannual variability and trends in vegetation greenness. Specifically, the new *data.table* summarizes for each sample location the trend slope, intercept, Kendall's tau, and p-value, as well as total absolute and relative change in vegetation greenness and other information (e.g., number of years with observations). The console message summarizes the mean (±1 SD) relative change in vegetation greenness across all sample locations, as well as the percentage of samples sites that greened, browned, or had no trend based on a user-specified critical value (default $\alpha = 0.10$). The multi-panel figure provides (a) a histogram of relative change in vegetation greenness among sample locations and (b) a time series of annual mean (± 1 SE) vegetation greenness for sample locations that greened, browned, or had no trend.

Example application: Vegetation greenness trends for a landscape on Disko Island

Here we provide an example analysis of interannual changes in vegetation greenness from 2000 to 2020 across a ~4 km^2 study area on Disko Island off the western coast of Greenland (Figure 2). The study area (approximate center 69.27°N, 53.46°W) is located on the eastern slopes of the Blæsedalen valley just east of Qeqertarsuaq (Godhavn). The close proximity of the valley to the University of Copenhagen's Arctic Station has made the area subject to much ecological and geological research, including multiple long-term monitoring projects and experiments (https://arktiskstation.ku.dk). Climatically, the site lies within the transition zone between the low and high Arctic, with basaltic soils on discontinuous permafrost (Xu et al. 2021) covered by erect dwarf shrub tundra (Walker et al. 2002). We characterize annual maximum vegetation greenness using the Normalized Difference Vegetation Index ($NDVI_{max}$) derived from Landsat satellite observations. Landsat $NDVI_{max}$ relates to vegetation productivity and aboveground biomass in tundra ecosystems (Johansen and Tømmervik 2014, Berner et al. 2018, Berner et al. 2020). Here, we focus on the period from 2000 to 2020 because there was limited Landsat data available prior to 2000 in this region, as shown below. We provide the scripts associated with this example as supplemental files and in this section guide the reader through the analysis code with example output figures and tables that are generated by the *lsatTS* functions (excluding Figure 2).

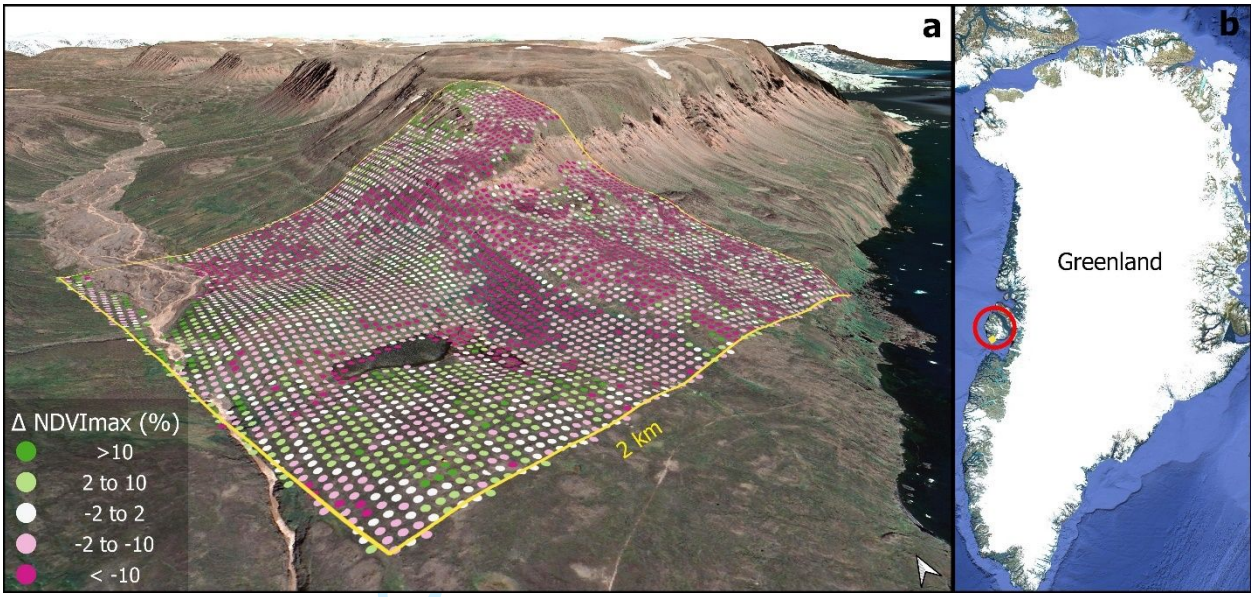

Figure 2. (a) Relative changes in Landsat annual maximum NDVI ($NDVI_{max}$) from 2000 to 2020 across the study area on Disko Island. (b) Location of study area off the western coast of Greenland. Figure created using QGIS (v3.20; QGIS.org 2021). Background imagery from Google Satellite © 2022 CNES / Airbus used with fair use permission. Underlying digital elevation model from the U.S. National Snow and Ice Data Center (Howat et al. 2014, Howat et al. 2015).

Part 1: Export Landsat time series from Google Earth Engine

First the user needs to export Landsat time series for sample locations in the study area using GEE (Code Box 1). For this they need to prepare the environment, set the boundaries of the study area and then retrieve the Landsat pixel center coordinates using the *lsat_get_pixel_centers()* function. Next, the Landsat records are exported for the pixel center locations using *lsat_export_ts()*. Here, we choose to export only Landsat observations in the between day of year 152 (beginning of June) and 273 (end of September). The user then waits for GEE to finish the exports. Progress can be monitored using the GEE task manager in the web browser (https://code.earthengine.google.com/tasks) or on the R console, using the *ee_monitoring()* function provided by *rgee*. For the example, it took ~2 days to export the 19 files (totaling ~692 MB) associated with this example analysis. The CSV files containing the raw exports then need to be copied from the user's Google Drive to the local machine that will carry out the subsequent processing using *lsatTS*. The files can be copied manually or using the *ee_drive_to_local()* function provided by *rgee*. Once the records are available locally, they need to be cleaned and processed into vegetation index time series as detailed in the next section.

Code Box 1: Export Landsat time series from Google Earth Engine

```
# Load required R packages
require(lsatTS)
require(rgee)
require(sf)
require(ggplot2)
require(data.table)

```

```
# Initialize Google Earth Engine
ee_Initialize()

# Create sf polygon of the study area
aoi.poly <- st_polygon(list(matrix(
  c(-332950,-2243300,
    -334950,-2243300,
    -334950,-2245300,
    -332950,-2245300,
    -332950,-2243300),
  ncol = 2,
  byrow = T)))

# Transform polygon to WGS84 lat long
aoi.poly <- aoi.poly %>%
  st_sfc(crs = 3413) %>%
  st_transform(crs = 4326) %>%
  st_as_sf()

# Get the central coordinates for each of the 4557 Landsat pixels in study area
aoi.pts <- lsat_get_pixel_centers(aoi.poly)

# Export summer Landsat surface reflectance measurements for each pixel to a folder
# called "earth_engine/lsat_disko" on the user's Google Drive.
lsat_export_ts(
  pixel_coords_sf = aoi.pts,
  startJulian = 152,
  endJulian = 273,
  prefix = 'disko',
  drive_export_dir = 'earth_engine/lsat_disko')
```

Part 2: Derive vegetation greenness time series from the raw Landsat data

To derive the vegetation greenness time series from the raw exports of Landsat time series, the records first need to be imported to R as a *data.table* object, re-formatted using *lsat_general_prep()* and cleaned with *lsat_clean_data()* to filter out clouds, snow, and water, as well as radiometric and geometric errors (Code Box 2). For the study area on Disko Island, *lsat_clean_data()* removed 1,817,683 of 2,452,693 observations (74.11%) in the data cleaning process. The availability of Landsat observations for all point locations ("sample.ids") in the remaining dataset can then be visualized using *lsat_summarize_data_avail*(). In the case of the pixel centers across the study area on Disko Island, the number of observations is poor before the year 2000, as highlighted by the graph that is automatically generated by the function (Figure 3). Therefore, we later limit the analysis of vegetation greenness to the years between 2000 and 2020. Finally, the NDVI is calculated using the *lsat_calc_spec_index()*. The dataset is then ready for the sensor cross-calibration and phenological modelling.

Code Box 2: Derived vegetation greenness time series from the raw Landsat data

```
# Import CSV exported with GEE as data.table
data.files <- list.files('~/earth_engine/lsat_disko', full.names = T)
lsat.dt <- do.call("rbind", lapply(data.files, fread))
```

```
# (Re-)format the imported raw data
lsat.dt <- lsat_general_prep(lsat.dt)

# Clean data by filtering clouds, snow, and water, as well as radiometric and
geometric errors.
lsat.dt <- lsat_clean_data(lsat.dt)

# Summarize the availability of Landsat data for each pixel
lsat_summarize_data_avail(lsat.dt)

# Compute the Normalized Difference Vegetation Index (NDVI)
lsat.dt <- lsat_calc_spec_index(lsat.dt, si = 'ndvi')
```

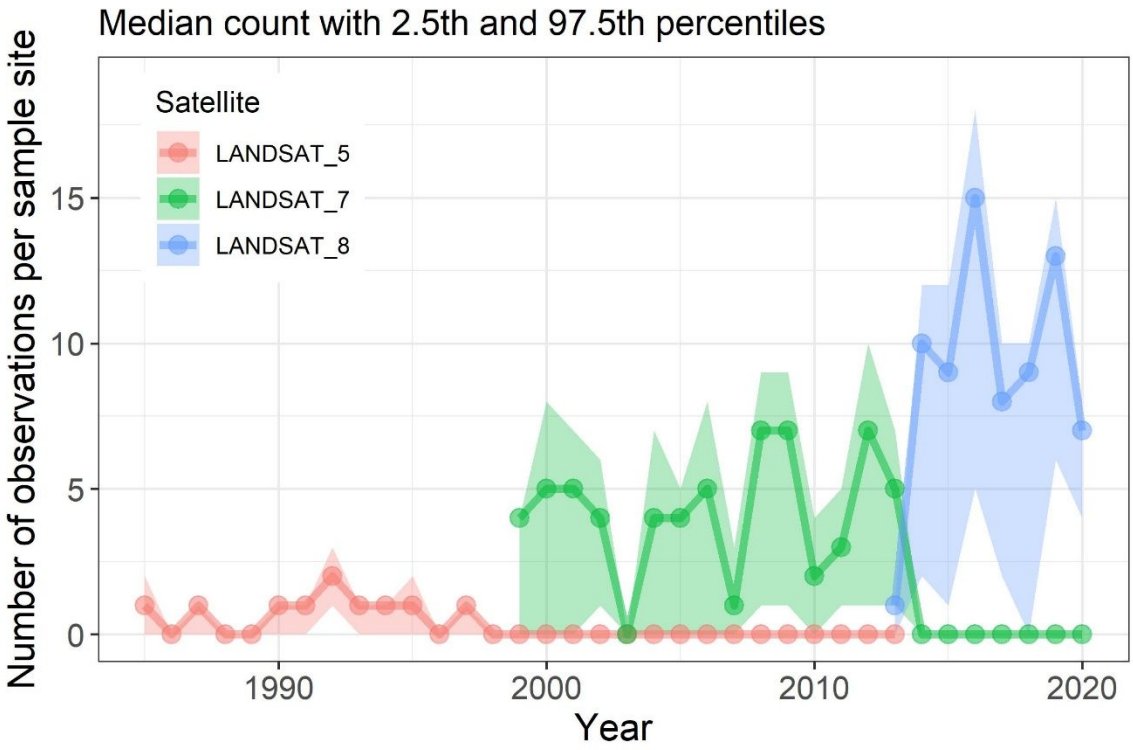

Figure 3. Availability of quality-screened Landsat observations across years for sample locations in the study area on Disko Island as returned by the l*sat_summarize_data_avail()* function. Summaries are based on observations acquired between day-of-year 152 (beginning of June) and 273 (end of September). Note the limited availability of observations before the year 2000. Lines with points denote median counts while shaded error bands encompass the 2.5th to 97.5th percentiles of counts among sample locations.

Part 3: Cross-sensor calibration and phenological modelling

The derived NDVI time series need to be calibrated across the different Landsat sensors, and then $NDVI_{max}$ estimated using the phenological modelling approach (Code Box 3). We start by cross-calibrating the time series using *lsat_calibrate_rf()*. As the number of observations in the Disko Island dataset is too small to calibrate the random forest models, we use the pre-processed dataset of high latitude observations included with *lsatTS*. The function saves the models in a

specified output directory and generates a series of graphs (Figure 4) and tabular data (Table 3) that help with evaluating model performance. As desired, the calibration reduced the median bias between the Landsat 7 observations and the Landsat 5 and 8 observations visually (Figure 4) and statistically (Table 3). Next, as a step towards estimating annual $NDVI_{max}$, we fit phenological models to the calibrated NDVI time series using *lsat_fit_phenological_curves()*. The function automatically returns a figure with Landsat observations and fitted phenological curves for nine random sample locations in the dataset (Figure 5). Each phenological curve characterizes the seasonal progression of NDVI using observations pooled over a multi-year period (here an 11 year moving window) and should be smooth and hump-shaped. Beware of phenological curves with long straight lines that could suggest inadequate seasonal distribution of data used when fitting the curves. Once the models are fitted, the summary statistics (including the estimated $NDVI_{max}$) are extracted using *lsat_summarize_growing_seasons()*. The *lsat_evaluate_phenological_max()* can be used to output a figure that allows for visually assessing the performance of modelled $NDVI_{max}$ (Figure 6). In the case of this Disko Island dataset, modeled estimates of $NDVI_{max}$ tend to be biased slightly low (~1%) when only one or two observations are available from a growing season (Figure 6), yet there were rarely such few observations during the period from 2000 to 2020 (Figure 3). The final step following the cross-calibration and phenological modelling is the time series analysis.

Code Box 3: Cross-calibration and phenological modelling

```
# Cross-calibrate NDVI among sensors using random forest models
# Outputs in Figure 4 and Table 3.
Lsat.dt <- lsat_calibrate_rf(
  lsat.dt,
  band.or.si = 'ndvi',
  train.with.highlat.data = T,
  outdir = 'output/ndvi_xcal_smry/',
  overwrite.col = T)

# Fit phenological models (cubic splines) to time series at
# each sample location (Figure 5)
lsat.pheno.dt <- lsat_fit_phenological_curves(lsat.dt, si = 'ndvi')

# Summarize spectral characteristics for each growing season
lsat.gs.dt <- lsat_summarize_growing_seasons(lsat.pheno.dt, si = 'ndvi')

# Evaluate the estimates of annual maximum NDVI (Figure 6)
lsat.eval.dt <- lsat_evaluate_phenological_max(lsat.pheno.dt, si = 'ndvi')
```

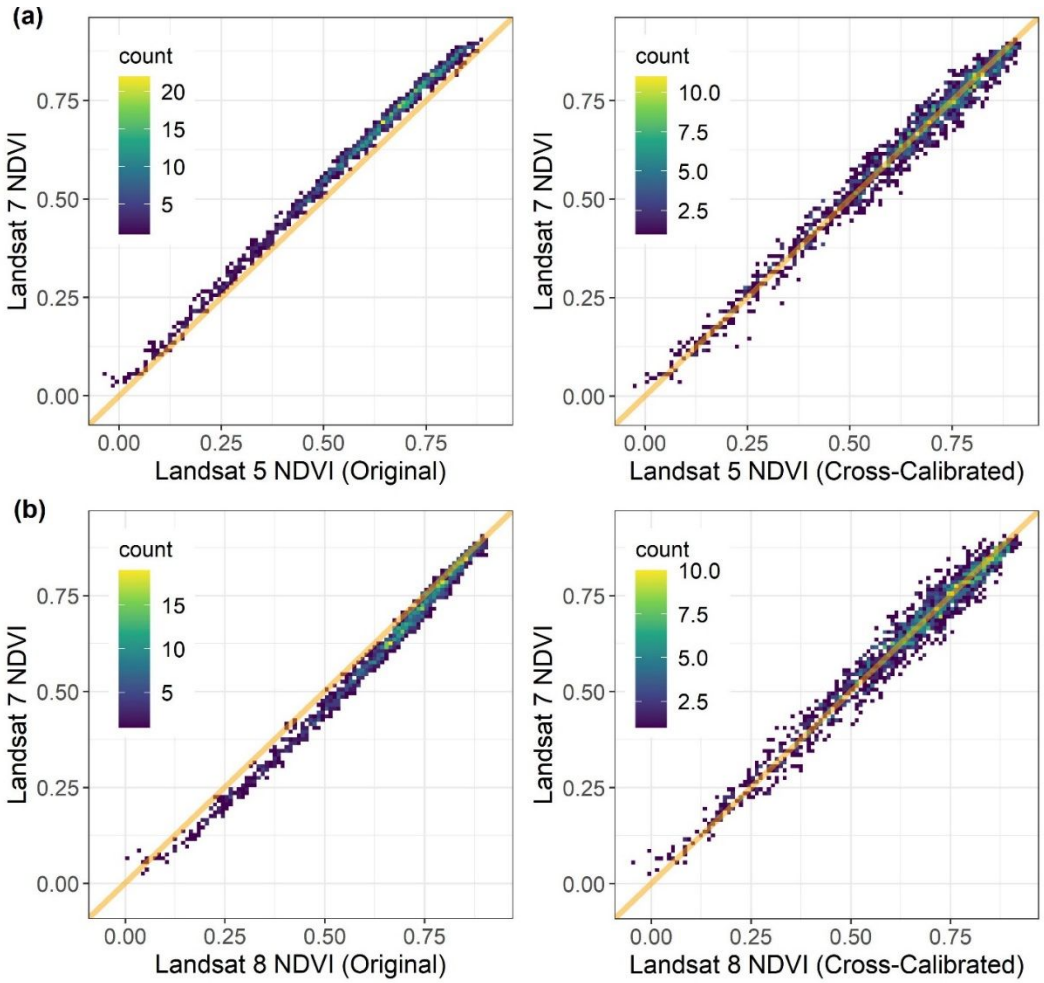

Figure 4. Relationships between Landsat 7 NDVI and both (a) Landsat 5 NDVI and (b) Landsat 8 NDVI using (left panels) original data and (right panels) data that were cross-calibrated with random forest models. Each point is a sample location from the Arctic – Boreal domain where there were temporally overlaps measurements from pairs Landsat satellites. Orange diagonal lines depict 1:1 relationships. Model performance metrics are provided in Table 3. Note that cross-calibration substantially reduces biases between sensors but does increase scatter.

Table 3. Summary of original biases, performance of random forest models for cross-sensor calibration, and post-calibration biases in NDVI between Landsat 7 ETM and either Landsat 5 TM or Landsat 8 ETM+. Error metrics were derived internally by the random forest using out-of-bag (OOB, i.e., withheld) data and further assessed using cross-validation, which yielded nearly identical results albeit with further information on post-calibration biases.

Satellite sensor	Original Data		OOB Error Metrics			Cross-Validated Error Metrics				
	Median bias	Median % bias	r^2	RMSE	N	r^2	RMSE	N	Median bias	Median % bias
Landsat 5 TM	-0.04	-6.1	0.98	0.03	4315	0.98	0.03	1438	+0.001	+0.1
Landsat 8 ETM+	+0.03	+4.6	0.97	0.03	4881	0.97	0.03	1627	-0.001	-0.1

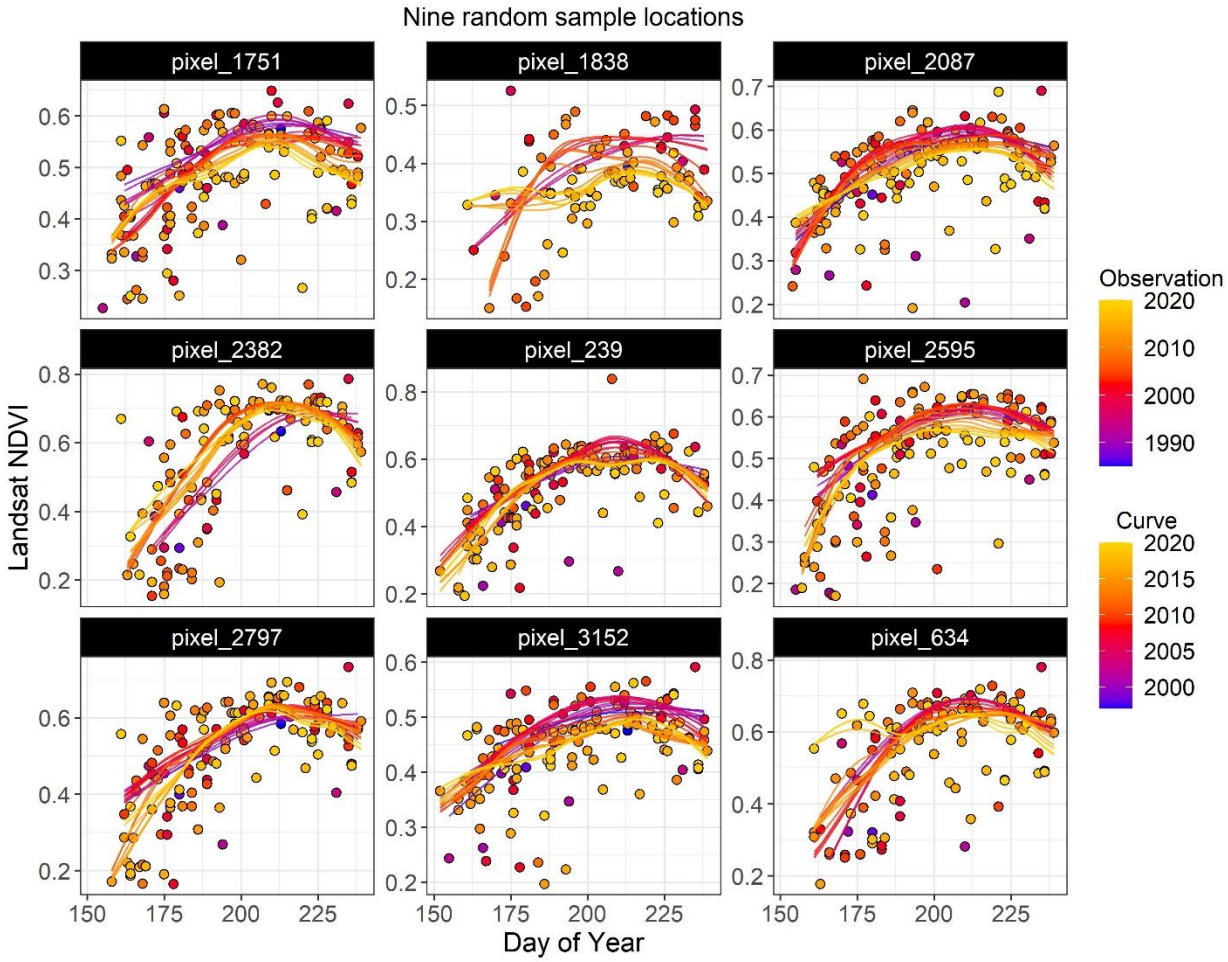

Figure 5. Seasonal progression of Landsat NDVI and phenological curves for nine random sample locations from the study area on Disko Island. Each point is an observation sorted by the day of year it was acquired and colored by the year of acquisition. Each phenological curve was fit to observations pooled over an 11-year window centered on each focal year.

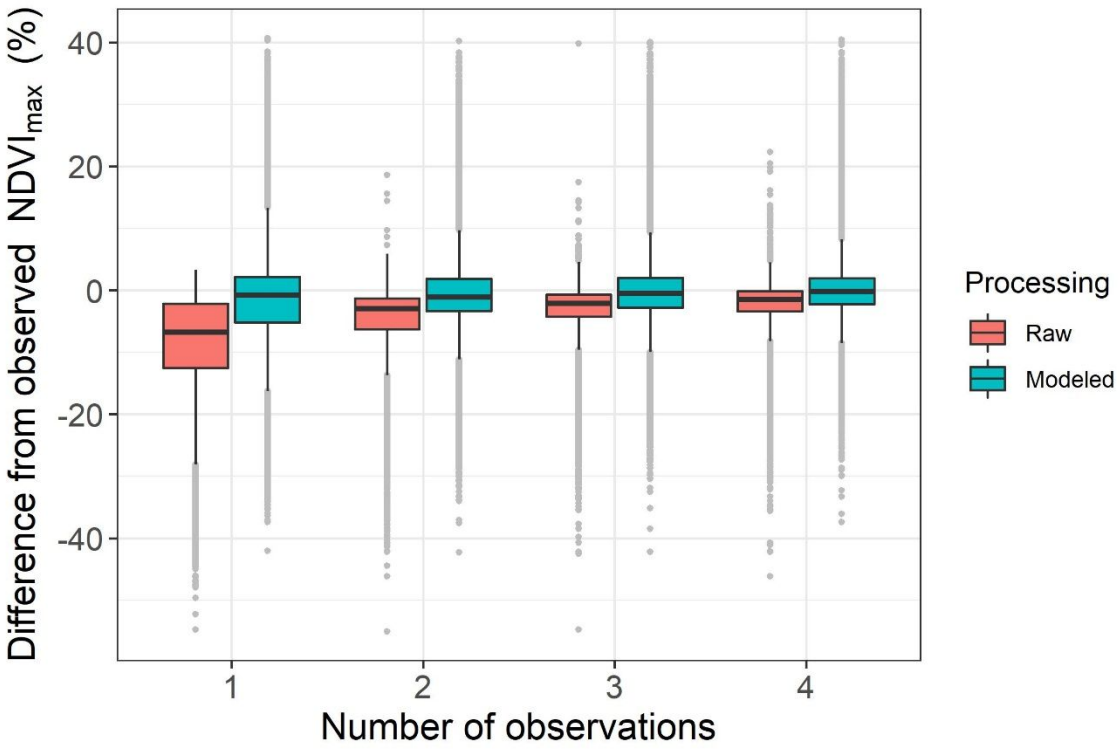

Figure 6. Raw estimates of annual maximum NDVI ($NDVI_{max}$) are biased low when few Landsat observations are available from a given growing season, whereas phenologically modeled estimates of $NDVI_{max}$ are minimally impacted by the availability of observations. The figure summarizes how raw and modeled estimate of $NDVI_{max}$ differ from observed $NDVI_{max}$ based on number of observations, as determined using *lsat_evaluate_phenological_max()*.

Part 3: Analyze vegetation greenness time series

Finally, the trend in the $NDVI_{max}$ across years for each sample location (pixel center) in our study area on Disko Island is calculated using the *lsat_calc_trend()* function (Code Box 4). Note how we use the "yrs" argument to restrict the time series analysis to the years between 2000-2020 to avoid using the low number of observations in the record prior the turn of the millennium. Figure 7 shows a histogram of precent change in $NDVI_{max}$ across the study area and a time series of annual mean $NDVI_{max}$ by trend category, both of which are generated by the function. These figures indicate extensive browning across the study area in recent decades.

Code Box 4: Analyze vegetation greenness time series

```
# Compute temporal trend in NDVImax (Figure 7)
lsat.trend.dt <- lsat_calc_trend(lsat.gs.dt, si = 'ndvi.max', yrs = 2000:2020)
```

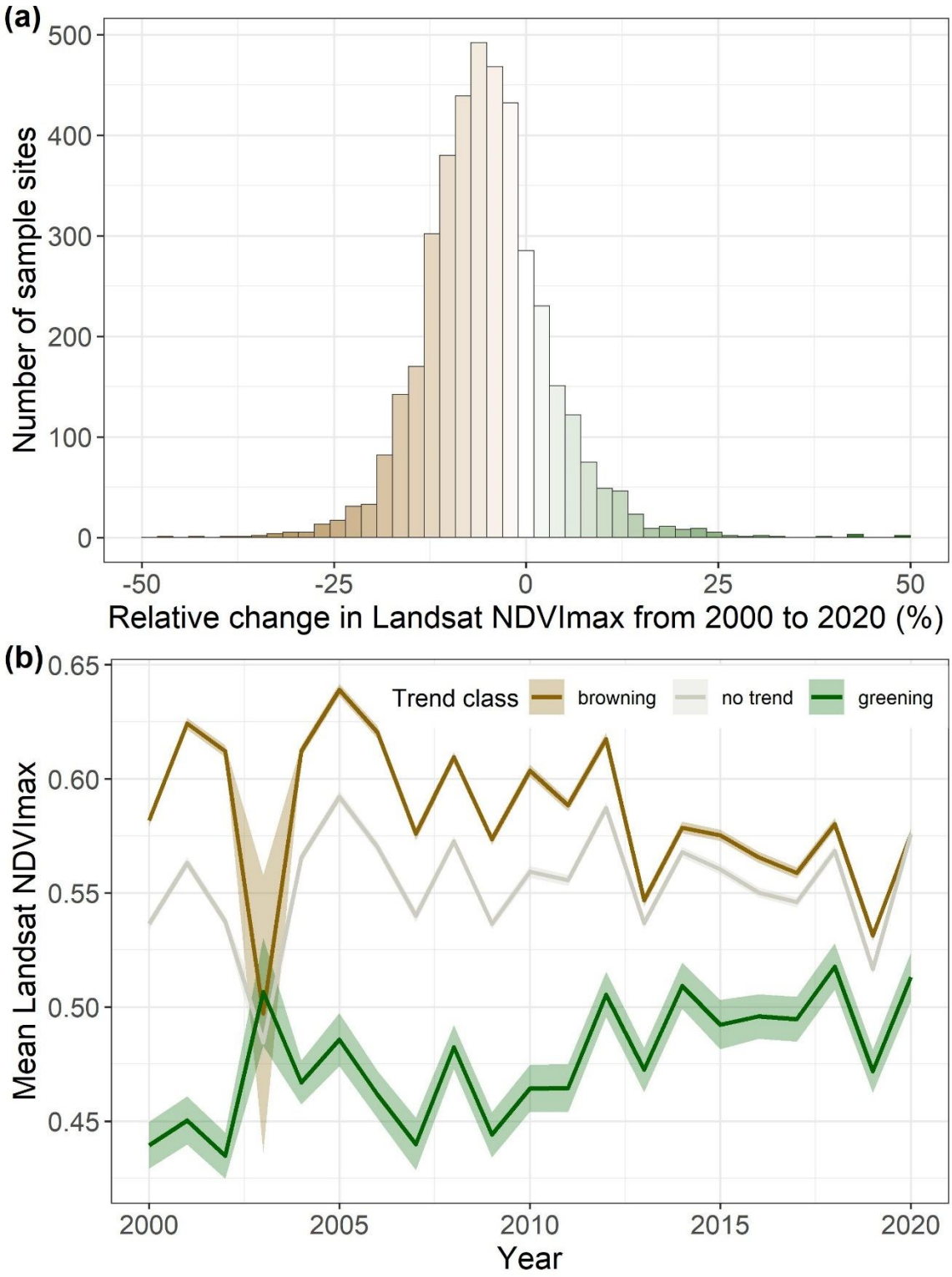

Figure 7. (a) Histogram of relative change in Landsat $NDVI_{max}$ from 2000 to 2020 among sample locations in the study area on Disko Island. Landsat $NDVI_{max}$ decreased (browned) across much of the study area over the past two decades. (b) Annual mean Landsat $NDVI_{max}$ from 2000 to 2020 for sample locations grouped by their concomitant temporal trend. Trends were assessed

for each sample location by removing temporal autocorrelation and then applying a Mann-Kendall trend test. Error bands depict ±1 standard error.

Results from the example study

This example analysis showed that from 2000 to 2020, annual maximum vegetation greenness (i.e., $NDVI_{max}$) systematically decreased ($\alpha = 0.10$; browned) across 51% of the study area on Disko Island, whereas vegetation greenness systematically increased ($\alpha = 0.10$; greened) across 3% of this study area (Figure 2a and 8). There were no systematic changes across the remaining 46% of the study area (Figure 2a and 8). Overall, vegetation greenness decreased by an average of 5.7 ± 8.4% (± 1 SD) during this period. The predominance of browning in this study area contrasts with widespread greening in the Arctic (Myers-Smith et al. 2020, Frost et al. 2021), where Landsat observations indicate that average Arctic vegetation greenness increased 3.9% from 2000 to 2020 (Berner et al. 2020, Mekonnen et al. 2021). Nevertheless, browning in this study area is broadly consistent with findings from recent pan-Arctic analyses using Landsat (Berner et al. 2020) and MODIS (Frost et al. 2021) satellite data that show regional browning in southwestern Greenland. The causes of browning in southwestern Greenland warrant further investigation but are potentially linked to hotter and dried conditions suppressing shrub and other vegetation growth and, in some areas, with defoliation from moths (*Eurois occulta*) or browsing by muskoxen (*Ovibos moschatus*) (Forchhammer 2017, Gamm et al. 2018, Prendin et al. 2020). This analysis demonstrates a general workflow that can be used to not only explore long-term changes in vegetation greenness across focal landscapes, but also to perform sample-based analyses across large geographic domains.

Conclusion

The *lsatTS* package for R facilitates the extraction and processing of Landsat surface reflectance time series, as well as generating and analyzing metrics of vegetation greenness and other spectral indices. We demonstrated the functionality of this software by analyzing changes in vegetation greenness across a tundra landscape on Disko Island off the west coast of Greenland, but would like to highlight that these tools are also well suited for sample-based analyses of vegetation dynamics across large geographic regions such as whole terrestrial biomes (e.g., Berner et al. 2020, Berner and Goetz 2022). To date, *lsatTS* has been used for ecological studies focused on Arctic tundra and boreal forest, but many of the functions could be used for studies focused on lower latitude ecosystems, especially ecosystems with a single growing season. Overall, this software provides a suite of functions to enable broader use of Landsat satellite data for assessing and monitoring Earth's land surface over the past four decades in a sample-based framework suitable for local to global geographic extents.

References

Appelhans, T., F. Detsch, C. Reudenbach, and S. Woellauer. 2021. mapview: Interactive Viewing of Spatial Data in R. R package version 2.10.0. https://CRAN.R-project.org/package=mapview.

Aybar, C., Q. Wu, L. Bautista, R. Yali, and A. Barja. 2020. rgee: An R package for interacting with Google Earth Engine. Journal of Open Source Software **5**:2272.

Bache, S. M., and H. Wickham. 2020. magrittr: A Forward-Pipe Operator for R. R package version 2.0.1. https://CRAN.R-project.org/package=magrittr.

Badgley, G., C. B. Field, and J. A. Berry. 2017. Canopy near-infrared reflectance and terrestrial photosynthesis. Science Advances **3**:e1602244.

Bengtsson, H. 2021. R.utils: Various Programming Utilities. R package version 2.11.0. https://CRAN.R-project.org/package=R.utils.

Berner, L. T., and S. J. Goetz. 2022. Satellite observations document trends consistent with a boreal forest biome shift. Global Change Biology **00**:1-18.

Berner, L. T., P. Jantz, K. D. Tape, and S. J. Goetz. 2018. Tundra plant aboveground biomass and shrub dominance mapped across the North Slope of Alaska. Environmental Research Letters **13**:035002.

Berner, L. T., R. Massey, P. Jantz, B. C. Forbes, M. Macias-Fauria, I. H. Myers-Smith, T. Kumpula, G. Gauthier, L. Andreu-Hayles, B. Gaglioti, P. J. Burns, P. Zetterberg, R. D'Arrigo, and S. J. Goetz. 2020. Summer warming explains widespread but not uniform greening in the Arctic tundra biome. Nature communications **11**:4621.

Boyd, M. A., L. T. Berner, P. Doak, S. J. Goetz, B. M. Rogers, D. Wagner, X. J. Walker, and M. C. Mack. 2019. Impacts of climate and insect herbivory on productivity and physiology of trembling aspen (Populus tremuloides) in Alaskan boreal forests. Environmental Research Letters **14**:085010.

Boyd, M. A., L. T. Berner, A. C. Foster, S. J. Goetz, B. M. Rogers, X. J. Walker, and M. C. Mack. 2021. Historic declines in growth portend trembling aspen death during a contemporary leaf miner outbreak in Alaska. Ecosphere **12**:e03569.

Breiman, L. 2001. Random Forests. Machine Learning **45**:5-32.

Bronaugh, D., and A. Werner. 2019. zyp: Zhang + Yue-Pilon trends package. R package version 0.10-1.1. https://CRAN.R-project.org/package=zyp.

Camps-Valls, G., M. Campos-Taberner, Á. Moreno-Martínez, S. Walther, G. Duveiller, A. Cescatti, M. D. Mahecha, J. Muñoz-Marí, F. J. García-Haro, and L. Guanter. 2021. A unified vegetation index for quantifying the terrestrial biosphere. Science Advances **7**:eabc7447.

Csárdi, G. 2021. crayon: Colored Terminal Output. R package version 1.4.2. https://CRAN.R-project.org/package=crayon.

dos Santos, A. 2017. landsat8: Landsat 8 Imagery Rescaled to Reflectance, Radiance and/or Temperature. R package version 0.1-10. https://CRAN.R-project.org/package=landsat8.

Dowle, M., and A. Srinivasan. 2021. data.table: Extension of `data.frame`. R package version 1.14.2. https://CRAN.R-project.org/package=data.table.

Forchhammer, M. 2017. Sea-ice induced growth decline in Arctic shrubs. Biology Letters **13**.

Frost, G. V., M. J. Macander, U. S. Bhatt, H. E. Epstein, L. T. Berner, J. W. Bjerke, B. C. Forbes, S. J. Goetz, M. J. Lara, T. Park, G. K. Phoenix, M. K. Raynolds, H. Tømmervik, and D. A. Walker. 2021. Tundra greenness [in "State of the Climate in 2020"]. Bulletin of the American Meteorological Society **102**:S290–S292.

Gaglioti, B., L. T. Berner, B. M. Jones, K. M. Orndahl, A. P. Williams, L. Andreu-Hayles, R. D'Arrigo, S. J. Goetz, and D. H. Mann. 2021. Tussocks enduring or shrubs greening: Alternate responses to changing fire regimes in the Noatak River Valley, Alaska. Journal of Geophysical Research: Biogeosciences **126**:e2020JG006009.

Gamm, C. M., P. F. Sullivan, A. Buchwal, R. J. Dial, A. B. Young, D. A. Watts, S. M. Cahoon, J. M. Welker, and E. Post. 2018. Declining growth of deciduous shrubs in the warming climate of continental western Greenland. Journal of Ecology **106**:640-654.

Gao, B.-C. 1996. NDWI—A normalized difference water index for remote sensing of vegetation liquid water from space. Remote Sensing of Environment **58**:257-266.

Gitelson, A. A. 2004. Wide dynamic range vegetation index for remote quantification of biophysical characteristics of vegetation. Journal of plant physiology **161**:165-173.

Gitelson, A. A., and M. N. Merzlyak. 1998. Remote sensing of chlorophyll concentration in higher plant leaves. Advances in Space Research **22**:689-692.

Goetz, S. J., and S. D. Prince. 1999. Modelling Terrestrial Carbon Exchange and Storage: Evidence and Implications of Fuctional Convergence in Light-use Efficiency. Advances in Ecological Research **28**:57-92.

Gorelick, N., M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore. 2017. Google Earth Engine: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment **202**:18-27.

Goslee, S. 2011. Analyzing remote sensing data in R: The Landsat Package. The Journal of Statistial Software **43**.

Hansen, M. C., P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R. G. Townshend. 2013. High-Resolution Global Maps of 21st-Century Forest Cover Change. science **342**:850.

Hardisky, M., V. Klemas, and M. Smart. 1983. The influence of soil salinity, growth form, and leaf moisture on the spectral radiance of Spartina alterniflora. Photogrammetric Engineering & Remote Sensing **49**:77-83.

Henry, L., and H. Wickham. 2020. purrr: Functional Programming Tools. R package version 0.3.4. https://CRAN.R-project.org/package=purrr.

Howat, I., A. Negrete, and B. Smith. 2015. MEaSUREs Greenland Ice Mapping Project (GIMP) Digital Elevation Model, Version 1. NASA National Snow and Ice Data Center Distributed Active Archive Center. doi: https://doi.org/10.5067/NV34YUIXLP9W. [2021-11-23], Boulder, Colorado USA.

Howat, I. M., A. Negrete, and B. E. Smith. 2014. The Greenland Ice Mapping Project (GIMP) land classification and surface elevation data sets. The Cryosphere **8**:1509-1518.

Huete, A., K. Didan, T. Miura, E. P. Rodriguez, X. Gao, and L. G. Ferreira. 2002. Overview of the radiometric and biophysical performance of the MODIS vegetation indices. Remote Sensing of Environment **83**:195-213.

Huete, A. R. 1988. A soil-adjusted vegetation index (SAVI). Remote Sensing of Environment **25**:295-309.

Jiang, Z., A. R. Huete, K. Didan, and T. Miura. 2008. Development of a two-band enhanced vegetation index without a blue band. Remote Sensing of Environment **112**:3833-3845.

Johansen, B., and H. Tømmervik. 2014. The relationship between phytomass, NDVI and vegetation communities on Svalbard. International Journal of Applied Earth Observation and Geoinformation **27**:20-30.

Ju, J., and J. G. Masek. 2016. The vegetation greenness trend in Canada and US Alaska from 1984–2012 Landsat data. Remote Sensing of Environment **176**:1-16.

Kassambara, A. 2020. ggpubr: 'ggplot2' Based Publication Ready Plots. R package version 0.4.0. https://CRAN.R-project.org/package=ggpubr.

Key, C. H., and N. C. Benson. 1999. The Normalized Burn Ratio (NBR): A Landsat TM radiometric measure of burn severity. United States Geological Survey, Northern Rocky Mountain Science Center.(Bozeman, MT).

Marsett, R. C., J. Qi, P. Heilman, S. H. Biedenbender, M. C. Watson, S. Amer, M. Weltz, D. Goodrich, and R. Marsett. 2006. Remote sensing for grassland management in the arid southwest. Rangeland Ecology & Management **59**:530-540.

McFeeters, S. K. 1996. The use of the Normalized Difference Water Index (NDWI) in the delineation of open water features. International Journal of Remote Sensing **17**:1425-1432.

Mekonnen, Z. A., W. J. Riley, L. T. Berner, N. J. Bouskill, M. S. Torn, G. Iwahana, A. L. Breen, I. H. Myers-Smith, M. G. Criado, Y. Liu, E. S. Euskirchen, S. J. Goetz, M. C. Mack, and R. F. Grant. 2021. Arctic tundra shrubification: a review of mechanisms and impacts on ecosystem carbon balance. Environmental Research Letters **16**:053001.

Merzlyak, M. N., A. A. Gitelson, O. B. Chivkunova, and V. Y. Rakitin. 1999. Non-destructive optical detection of pigment changes during leaf senescence and fruit ripening. Physiologia plantarum **106**:135-141.

Myers-Smith, I. H., J. T. Kerby, G. K. Phoenix, J. W. Bjerke, H. E. Epstein, J. J. Assmann, C. John, L. Andreu-Hayles, S. Angers-Blondin, P. S. A. Beck, L. T. Berner, U. S. Bhatt, A. D. Bjorkman, D. Blok, A. Bryn, C. T. Christiansen, J. H. C. Cornelissen, A. M. Cunliffe, S. C. Elmendorf, B. C. Forbes, S. J. Goetz, R. D. Hollister, R. de Jong, M. M. Loranty, M. Macias-Fauria, K. Maseyk, S. Normand, J. Olofsson, T. C. Parker, F.-J. W. Parmentier, E. Post, G. Schaepman-Strub, F. Stordal, P. F. Sullivan, H. J. D. Thomas, H. Tømmervik, R. Treharne, C. E. Tweedie, D. A. Walker, M. Wilmking, and S. Wipf. 2020. Complexity revealed in the greening of the Arctic. Nature Climate Change **10**:106-117.

National Academies of Sciences. 2018. Thriving on Our Changing Planet: A Decadal Strategy for Earth Observation from Space. The National Academies Press, Washington, DC.

Pastick, N. J., M. T. Jorgenson, S. J. Goetz, B. M. Jones, B. K. Wylie, B. J. Minsley, H. Genet, J. F. Knight, D. K. Swanson, and J. C. Jorgenson. 2019. Spatiotemporal remote sensing of ecosystem change and causation across Alaska. Global Change Biology **25**:1171-1189.

Pebesma, E. J. 2018. Simple features for R: standardized support for spatial vector data. The R Journal **10**:439-446.

Pekel, J.-F., A. Cottam, N. Gorelick, and A. S. Belward. 2016. High-resolution mapping of global surface water and its long-term changes. Nature **540**:418-422.

Prendin, A. L., M. Carrer, M. Karami, J. Hollesen, N. Bjerregaard Pedersen, M. Pividori, U. A. Treier, A. Westergaard-Nielsen, B. Elberling, and S. Normand. 2020. Immediate and carry-over effects of insect outbreaks on vegetation growth in West Greenland assessed from cells to satellite. Journal of Biogeography **47**:87-100.

QGIS.org. 2021. QGIS Geographic Information System. QGIS Association. http://www.qgis.org.

R Core Team. 2021. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.

Rock, B., J. Vogelmann, D. Williams, A. Vogelmann, and T. Hoshizaki. 1986. Remote detection of forest damage. BioScience **36**:439-445.

Rouse, J., R. Haas, J. Schell, and D. Deering. 1974. Monitoring vegetation systems in the Great Plains with ERTS. NASA special publication **351**:309-317.

Roy, D. P., V. Kovalskyy, H. K. Zhang, E. F. Vermote, L. Yan, S. S. Kumar, and A. Egorov. 2016. Characterization of Landsat-7 to Landsat-8 reflective wavelength and normalized difference vegetation index continuity. Remote Sensing of Environment **185**:57-70.

Sexton, J. O., X.-P. Song, M. Feng, P. Noojipady, A. Anand, and C. Huang. 2013. Global, 30-m resolution continuous fields of tree cover: landsat-based rescaling of MODIS vegetation continuous fields with lidar-based estimates of error. Int J Digit Earth **6**.

Tucker, C. J. 1979. Red and photographic infrared linear combinations for monitoring vegetation. Remote Sensing of Environment **8**:127-150.

Verdonen, M., L. T. Berner, B. C. Forbes, and T. Kumpula. 2020. Periglacial vegetation dynamics in Arctic Russia: decadal analysis of tundra regeneration on landslides with time series satellite imagery. Environmental Research Letters **15**:105020.

Walker, D. A., W. A. Gould, H. A. Maier, and M. K. Raynolds. 2002. The Circumpolar Arctic Vegetation Map: AVHRR-derived base maps, environmental controls, and integrated mapping procedures. International Journal of Remote Sensing **23**:4551-4570.

Walker, X. J., H. D. Alexander, L. T. Berner, M. A. Boyd, M. M. Loranty, S. M. Natali, and M. C. Mack. 2021. Positive response of tree productivity to warming is reversed by increased tree density at the Arctic tundra-taiga ecotone. Canadian Journal of Forest Research **51**:1323-1338.

Wang, J. A., and M. A. Friedl. 2019. The role of land cover change in Arctic-Boreal greening and browning trends. Environmental Research Letters **14**:125007.

Wickham, H. 2016. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlang New York.

Wickham, H. 2019. stringr: Simple, Consistent Wrappers for Common String Operations. R package version 1.4.0. https://CRAN.R-project.org/package=stringr.

Wickham, H. 2021. tidyr: Tidy Messy Data. R package version 1.1.4. https://CRAN.R-project.org/package=tidyr.

Wickham, H., R. Francois, H. Lionel, and K. Müller. 2021. dplyr: A Grammar of Data Manipulation. R package version 1.0.7. https://CRAN.R-project.org/package=dplyr.

Woodcock, C. E., R. Allen, M. Anderson, A. Belward, R. Bindschadler, W. Cohen, F. Gao, S. N. Goward, D. Helder, E. Helmer, R. Nemani, L. Oreopoulos, J. Schott, P. S. Thenkabail, E. F. Vermote, J. Vogelmann, M. A. Wulder, R. Wynne, and T. Landsat Sci. 2008. Free access to Landsat imagery. science **320**:1011-1011.

Wright, M. N., and A. Ziegler. 2017. Ranger: a fast implementation of random forests for high dimensional data in C++ and R. Journal of statistical software **77**:1-17.

Wulder, M. A., T. R. Loveland, D. P. Roy, C. J. Crawford, J. G. Masek, C. E. Woodcock, R. G. Allen, M. C. Anderson, A. S. Belward, W. B. Cohen, J. Dwyer, A. Erb, F. Gao, P. Griffiths, D. Helder, T. Hermosilla, J. D. Hipple, P. Hostert, M. J. Hughes, J. Huntington, D. M. Johnson, R. Kennedy, A. Kilic, Z. Li, L. Lymburner, J. McCorkel, N. Pahlevan, T. A. Scambos, C. Schaaf, J. R. Schott, Y. Sheng, J. Storey, E. Vermote, J. Vogelmann, J. C. White, R. H. Wynne, and Z. Zhu. 2019. Current status of Landsat program, science, and applications. Remote Sensing of Environment **225**:127-147.

Xu, W., A. Prieme, E. J. Cooper, M. A. Mörsdorf, P. Semenchuk, B. Elberling, P. Grogan, and P. L. Ambus. 2021. Deepened snow enhances gross nitrogen cycling among Pan-Arctic tundra soils during both winter and summer. Soil Biology and Biochemistry **160**:108356.

Yue, S., P. Pilon, B. Phinney, and G. Cavadias. 2002. The influence of autocorrelation on the ability to detect trend in hydrological series. Hydrological processes **16**:1807-1829.

Zeileis, A., and G. Grothendieck. 2005. zoo: S3 Infrastructure for Regular and Irregular Time Series. Journal of statistical software **14**:1-27.

Zhu, Z., S. Wang, and C. E. Woodcock. 2015. Improvement and expansion of the Fmask algorithm: cloud, cloud shadow, and snow detection for Landsats 4–7, 8, and Sentinel 2 images. Remote Sensing of Environment **159**:269-277.

Package 'lsatTS'

April 13, 2022

Title An R package to facilitate retrieval, cleaning, cross-calibration, and phenological modeling of Landsat time-series data

Version 1.0.0

Description This software package facilitates sample-based time series analysis of surface reflectance and spectral indices derived from sensors on the Landsat satellites. The package includes functions that enable extraction of the full Landsat record for point sample locations or small study regions using Google Earth Engine directly accessed from R. Moreover, the package includes functions for (1) rigorous data cleaning, (2) cross-sensor calibration with machine learning, (3) phenological modeling, and (4) time series analysis.

License Modified MIT

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports magrittr, dplyr, tidyr, rgee, sf, crayon, mapview, purrr, data.table, ggplot2, R.utils, stats, stringr, ggpubr, ranger, zoo, zyp

Depends R (>= 3.50)

R topics documented:

lsat.example.dt *Landsat surface reflectance for six sample sites in the Arctic*

Description

A dataset containing Landsat surface reflectance measurements and ancillary data for six sample sites in the Arctic. These data are used for the examples included with lsatTS.

Usage

```
lsat.example.dt
```

Format

A data.table with 5296 rows and 23 variables

Source

Generated using example code provided in lsatTS::lsat_export_ts()

lsat_calc_spec_index *Calculate spectral indices*

Description

This function computes some widely used spectral vegetation indices. Only one index can be computed at a time. Current indices include the: Normalized Difference Vegetation Index (NDVI; Rouse et al. 1974), kernel NDVI (kNDVI; Camp-Valls et al. 2020), Green NDVI (gNDVI; Gitelson and Merzlyak 1998), Soil Adjusted Vegetation Index (SAVI; Huete 1998), Wide Dynamic Range Vegetation Index (WDRVI; Gitelson 2004), Enhanced Vegetation Index (EVI; Huete et al. 2002), 2-band EVI (EVI2; Jiang et al. 2008), Near Infrared Vegetation Index (NIRv; Badgley et al. 2017), Moisture Stress Index (MSI; Rock et al. 1986), Normalized Difference Water Index (NDWI; McFeeters 1996), Normalized Difference Moisture Index (NDMI; Gao 1996), Normalized Burn Ratio (NBR, Key and Benson 1999), Normalized Difference Infrared Index (NDII; Hardisky et al. 1983), Plant Senescence Reflectance Index (PSRI; Merzlyak et al. 1999), and the Soil-Adjusted Total Vegetation Index (SATVI; Marsett et al. 2006).

Usage

```
lsat_calc_spec_index(dt, si)
```

Arguments

dt	Data.table containing surface reflectance data.
si	Character string specifying abbreviation of the desired spectral index.

Value

The input data.table with an appended column containing the spectral index.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt <- lsat_calc_spec_index(lsat.dt, 'ndvi')
lsat.dt
```

`lsat_calc_trend` *Calculate non-parametric vegetation greenness trends*

Description

This function evaluates and summarizes interannual trends in vegetation greenness for sample sites over a user-specificed time period. Potential interannual trends in vegetation greenness are assessed using Mann-Kendall trend tests and Theil-Sen slope indicators after prewhitening each time series. This trend assessment relies on the zyp.yuepilon() function from the zyp package, which provides further details.

Usage

```
lsat_calc_trend(
  dt,
  si,
  yrs,
  yr.tolerance = 1,
  nyr.min.frac = 0.66,
  sig = 0.1,
  legend.position = c(0.8, 0.2),
  legend.direction = "horizontal"
)
```

Arguments

`dt`	Data.table with columns including site, year, and the vegetation index of interest.
`si`	Spectral index for which to assess trend (e.g., NDVI).
`yrs`	A sequence of years over which to assess trends (e.g., 2000:2020).
`yr.tolerance`	The number of years that a site's first/last years of observations can differ from the start/end of the user-specified time period ('yrs') for a trend to be computed.
`nyr.min.frac`	Fraction of years within the time period for which observations must be available if a trend is to be computed.
`sig`	A p-value significance cutoff used to categories trends (e.g., 0.10)
`legend.position`	Legend position for output plot, specified as a vector with x and y values ranging from 0 to 1.
`legend.direction`	Legend direction for output plot, either "horizontal" or "vertical"

Value

Data.table with summary of temporal trends by site and a multi-panel figure with: (1) a histogram of relative changes in vegetation greenness among sample sites and (2) a time-series plot of mean vegetation greenness for sample sites grouped by trend category.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt <- lsat_calc_spec_index(lsat.dt, 'ndvi')
# lsat.dt <- lsat_calibrate_rf(lsat.dt, band.or.si = 'ndvi', write.output = F)
lsat.pheno.dt <- lsat_fit_phenological_curves(lsat.dt, si = 'ndvi')
lsat.gs.dt <- lsat_summarize_growing_seasons(lsat.pheno.dt, si = 'ndvi')
lsat.trend.dt <- lsat_calc_trend(lsat.gs.dt, si = 'ndvi.max', yrs = 2000:2020)
lsat.trend.dt
```

`lsat_calibrate_rf` *Cross-calibrate Landsat sensors using Random Forests models*

Description

There are systematic differences in spectral indices (e.g., NDVI) among Landsat 5, 7, and 8 (Landsat Collection 2). It is important to address these differences before assessing temporal trends in spectral data. Failure to address these differences can, for instance, introduce artificial positive trends into NDVI time-series that are based on measurements from multiple Landsat sensors (Ju and Masek 2016, Roy et al. 2016, Berner et al. 2020). This function cross-calibrates individual bands or spectral indices from Landsat 5/8 to match Landsat 7. Landsat 7 is used as a benchmark because it temporally overlaps with the other two sensors. Cross-calibration can only be performed on one band or spectral index at a time. The approach involves determining the typical reflectance at a sample during a portion of the growing season site using Landsat 7 and Landsat 5/8 data that were collected the same years. A Random Forest model is then trained to predict Landsat 7 reflectance from Landsat 5/8 reflectance. To account for potential seasonal and regional differences between sensors, the Random Forest models also include as covariates the midpoint of each 15-day period (day of year), the spatial coordinates of each sample sample, and potentially other use-specified variables. This approach is most suitable when working with data from 100s to preferably 1000s of sample samples.

The specific steps to cross-calibrating sensors include: (1) Identify the years when both Landsat 7 and Landsat 5/8 measured surface reflectance at a sample sample. (2) Pool the reflectance measurements across those years and compute 15-day moving median reflectance over the course of the growing season for each sensor and sampling sample. (3) Exclude 15-day periods with fewer than a specified number of measurements from both sets of sensors and then randomly select one remaining 15-day period from each sample sample. (4) Split the data into sets for model training and evaluation. (5) Train Random Forest models that predict Landsat 7 reflectance based on Landsat 5/8 reflectance. The models also account for potential seasonal and regional differences between sensors by including as covariates the midpoint of each 15-day period (day of year) and the spatial coordinates of each sampling sample. The models are trained using the ranger function from the ranger package (Wright and Ziegler, 2017). (6) Apply the fitted Random Forest models to cross-calibrate measurements.

See Berner et al. (2020) for a full description of the approach.

Usage

```
lsat_calibrate_rf(
  dt,
  band.or.si,
  doy.rng = 152:243,
  min.obs = 5,
  train.with.highlat.data = F,
  add.predictors = NULL,
  frac.train = 0.75,
  trim = T,
  overwrite.col = F,
  write.output = T,
  outfile.id = band.or.si,
  outdir = NA
)
```

Arguments

`dt`	Data.table containing the band or spectral index to cross-calibrate.
`band.or.si`	Character string matching the column name of the band or spectral index to cross-calibrate.
`doy.rng`	Sequence of numbers specifying the Days of Year to use for model development.
`min.obs`	Minimum number of paired, seasonally-matched observations from Landsat 7 and Landsat 5/8 required to include a sampling sample.
`train.with.highlat.data`	(True/False) Should the RF models be trained using an internal high-latitude dataset that sampled the Arctic and Boreal biomes?
`add.predictors`	Vector of additional predictors to use in the Random Forest models. These should be time-invariant and match column names.
`frac.train`	Fraction of samples to use for training the random forest models. The remaining samples are used for model cross-validation.
`trim`	(True/False) If true, then for each sample site the percent difference in spectral indices between satellites is determined. The lowest 2.5 and highest 97.5 percentiles are then trimmed. This is meant to reduce potential differences that are not directly attributable to the sensors, but rather to exogenous factors.
`overwrite.col`	(True/False) Overwrite existing column or (by default) append cross-calibrated data as a new column?
`write.output`	(True/False) Should RF models and evaluation content be written to disk? Either way, evaluation table and figure are printed in the console.
`outfile.id`	Identifier used when naming output files. Defaults to the input band, but can be specified if needed such as when performing Monte Carlo simulations.
`outdir`	Output directory (created if necessary) to which multiple files will be written. The files include: (1) fitted random forest models as R objects, (2) evaluation data in a csv file, (3) summary of model cross-validation in a csv file, and (4) multi-panel scatter plot comparing sensors pre- and post-calibration in jpeg format. If cross-calibrating both Landsat 5 and 8, then the function returns files for both sensors.

Value

The input data.table with an appended column titled band.xcal, where "band" is your specified band or spectral index.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt <- lsat_calc_spec_index(lsat.dt, 'ndvi')
# lsat.dt <- lsat_calibrate_rf(lsat.dt, band.or.si = 'ndvi', write.output = FALSE)
lsat.dt
```

`lsat_clean_data` *Clean Landsat surface reflectance data*

Description

This function enables users to filter out surface reflectance measurements that exhibit: (1) clouds, cloud shadows, snow, or water flagged by the CFMask algorithm; (2) surface water over the Landsat record; (2) impossibly high reflectance (>1.0) and abnormally low reflectance (<0.005); (3) scene cloud cover above a user-defined threshold; (4) geometric uncertainty above a user-defined threshold; (5) solar zenith angle above a user-defined threshold.

Usage

```
lsat_clean_data(
  dt,
  cloud.max = 80,
  geom.max = 30,
  sza.max = 60,
  filter.cfmask.snow = T,
  filter.cfmask.water = T,
  filter.jrc.water = T
)
```

Arguments

`dt`	Data.table generated by calling lsat_general_prep().
`cloud.max`	Maximum allowable cloud cover in Landsat scene (percentage).
`geom.max`	Maximum allowable geometric uncertainty (meters).
`sza.max`	Maximum allowable solar zenith angle (degrees).
`filter.cfmask.snow`	(TRUE/FALSE) Remove measurements with CFmask flag = snow.
`filter.cfmask.water`	(TRUE/FALSE) Remove measurements with CFmask flag = water.
`filter.jrc.water`	(TRUE/FALSE) Remove sample sites that were ever innundated based on the maximum surface water extent variable from the JRC Global Surface Water Dataset.

Value

A data.table that includes Landsat measurements that met the quality control criteria.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt
```

---

lsat_evaluate_phenological_max

Evaluate estimates of annual phenological maximum

Description

Assess how the number of annual Landsat measurements impacts estimates of annual maximum vegetation greenness derived from raw measurements and phenological modeling. The algorithm computes annual maximum vegetation greenness using site x years with a user-specific number of measurements and then compares these with estimates derived when using progressively smaller subsets of measurements. This lets the user determine the degree to which annual estimates of maximum vegetation greenness are impacted by the number of available measurements.

Usage

```
lsat_evaluate_phenological_max(
  dt,
  si,
  min.frac.of.max = 0.75,
  zscore.thresh = 3,
  min.obs = 6,
  reps = 10,
  outdir = NA
)
```

Arguments

`dt`	Data.table output from lsat_fit_phenological_curves().
`si`	Character string specifying the spectral index (SI) to evaluate (e.g., NDVI).
`min.frac.of.max`	Numeric threshold (0-1) that defines the "growing season" as the seasonal window when the phenological curves indicate the SI is within a specified fraction of the maximum SI. In other words, an observation is considered to be from the "growing season" when the SI is within a user-specified fraction of the curve-fit growing season maximum SI.
`zscore.thresh`	Numeric threshold specifying the Z-score value beyond which individual measurements are filtered before computing the maximum SI.
`min.obs`	Minimum number of measurements needed for a site x year to be included in the evaluation (Default = 10).
`reps`	Number of times to bootstrap the assessment (Default = 10).

`outdir` If desired, specify the output directory where evaluation data and figure should be written. If left as NA, then no output is only displayed in the console and not written to disk.

Value

A data.table and a figure summarizing how estimates of annnual maximum SI vary with the number of Landsat measurements made during each growing season.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt <- lsat_calc_spec_index(lsat.dt, 'ndvi')
# lsat.dt <- lsat_calibrate_rf(lsat.dt, band.or.si = 'ndvi', write.output = FALSE)
lsat.pheno.dt <- lsat_fit_phenological_curves(lsat.dt, si = 'ndvi')
lsat_evaluate_phenological_max(lsat.pheno.dt, si = "ndvi")
```

`lsat_export_ts` *Export reflectance time-series from the Landsat record using rgee*

Description

This function exports surface reflectance time series for a set of point-coordinates from the whole Landsat Collection 2 record using the Google Earth Engine. The resulting time-series can then be processed using the remainder of the lsatTS workflow.

For polygon geometries consider using lsat_get_pixel_centers() to generate pixel center coordinates for all pixels within a given polygon first.

Please note: Unlike other functions in this package, this function does NOT return the time-series as an object, instead it returns a list of the EE tasks issued for the export. The actual time-series are exported as CSV objects via the EE to the user's Google Drive. This way of exporting allows for a more efficient scheduling, larger exports, and does not require the R session to continue to run in the background while the requests are processed on the EE.

The progress of the exports can be monitored using the list of tasks returned in combination with ee_monitoring() from the rgee package, or simply by using the task overview in the web code-editor of the EE (https://code.earthengine.google.com).

Usage

```
lsat_export_ts(
  pixel_coords_sf,
  sample_id_from = "sample_id",
  chunks_from = NULL,
  this_chunk_only = NULL,
  max_chunk_size = 250,
  drive_export_dir = "lsatTS_export",
  file_prefix = "lsatTS_export",
  startJulian = 152,
```

```
  endJulian = 243,
  start_date = "1984-01-01",
  end_date = "today",
  BUFFER_DIST = 0,
  SCALE = 30,
  MASK_VALUE = 0
)
```

Arguments

pixel_coords_sf	Simple feature object of point coordinates for the sample.
sample_id_from	The column name that specifies the unique sample identifier in pixel_coords_sf (defaults to "sample_id" as generated by lsat_get_pixel_centers).
chunks_from	Column name in pixel_coords_sf to divide the exports into chunks. Over-rides chunk division by size (see max_chunk_size).
this_chunk_only	Name of a specific chunk to be exported. Useful for re-exporting a single chunk should the export fail for some reason.
max_chunk_size	Maximum number of sample coordinates to be exported in each chunk. Defaults to 250.
drive_export_dir	Folder on the user's Google Drive to export the records to. Defaults to "lsatTS_export".
file_prefix	Optional file_prefix for the exported files.
startJulian	Optional first day of year to extract for. Defaults to 152.
endJulian	Optional last day of year to extract for. Defaults to 243.
start_date	Optional extraction start date (as string, format YYYY-MM-DD). Defaults to "1984-01-01".
end_date	Optional extraction end date (as string, format YYYY-MM-DD). Defaults to today's date.
BUFFER_DIST	Buffer distance around sample coordinates. Wrapper for lsat_get_pixel_centers() to find all Landsat pixel centers around each point in pixel_coords_sf within the specified buffer distance (square)). Can be slow if the number of points is large. Defaults to 0 m.
SCALE	Scale for extraction. Defaults to 30 m nominal Landsat pixel size.
MASK_VALUE	Optional masking value for global surface water mask. Defaults to 0.

Value

List of initiated rgee tasks.

Author(s)

Jakob J. Assmann and Richard Massey

Examples

```
# Using sf, dplyr and rgee
library(sf)
library(dplyr)
library(rgee)

# Initialize EE
ee_Initialize()

# Generate test points
test_points_sf <- st_sfc(st_point(c(-149.6026, 68.62574)),
                         st_point(c(-149.6003, 68.62524)),
                         st_point(c(-75.78057, 78.87038)),
                         st_point(c(-75.77098, 78.87256)),
                         st_point(c(-20.56182, 74.47670)),
                         st_point(c(-20.55376, 74.47749)), crs = 4326) %>%
  st_sf() %>%
  mutate(sample_id = c("toolik_1",
                       "toolik_2",
                       "ellesmere_1",
                       "ellesmere_1",
                       "zackenberg_1",
                       "zackenberg_2"),
         region = c("toolik","toolik",
                    "ellesmere", "ellesmere",
                    "zackenberg", "zackenberg"))

# Export time-series using lsat_export_ts()
task_list <- lsat_export_ts(test_points_sf)

# Export time-series using with a chunk size of 2
task_list <- lsat_export_ts(test_points_sf, max_chunk_size = 2)

# Export time-series in chunks by column
task_list <- lsat_export_ts(test_points_sf, chunks_from = "region")
```

`lsat_fit_phenological_curves` *Characterize land surface phenology using spectral vegetation index time series*

Description

This function characterizes seasonal land surface phenology at each sample site using flexible cubic splines that are iteratively fit to time series of spectral vegetation indices (e.g., NDVI). This function facilitates estimating annual maximum NDVI and other spectral vegetation indices with lsat_summarize_growing_seasons(). For each site, cubic splines are iteratively fit to measurements pooled over years within a moving window that has a user-specified width. Each cubic spline is iteratively fit, with each iteration checking if there are outliers and, if so, excluding outliers and refitting. The function returns information about typical phenology at a sample site and about the relative phenological timing of each individual measuremenent. This function was designed for situations where the seasonal phenology is hump-shaped. If you are using a spectral index that is typically negative (e.g., Normalized Difference Water Index) then multiply the index by -1 before running

this function, then back-transform your index after running the lsat_summarize_growing_seasons() function.

Usage

```
lsat_fit_phenological_curves(
  dt,
  si,
  window.yrs = 11,
  window.min.obs = 20,
  si.min = 0.15,
  spar = 0.75,
  pcnt.dif.thresh = 30,
  weight = T,
  spl.fit.outfile = F,
  progress = T,
  test.run = F
)
```

Arguments

`dt`	Data.table with a multi-year time series a vegetation index.
`si`	Character string specifying the spectral index (e.g., NDVI) to use for determining surface phenology. This must correspond to an existing column in the data.table.
`window.yrs`	Number specifying the focal window width in years that is used when pooling data to fit cubic splines (use odd numbers).
`window.min.obs`	Minimum number of focal window observations necessary to fit a cubic spline.
`si.min`	Minimum value of spectral index necessary for observation to be used when fitting cubic splines. Defaults to 0.15 which for NDVI is about when plants are present. Note that si.min must be >= 0 because the underlying spline fitting function will error out if provided negative values.
`spar`	Smoothing parameter typically around 0.70 - 0.80 for this application. A higher value means a less flexible spline.
`pcnt.dif.thresh`	Allowable percent difference (0-100) between individual observations and fitted cubic spline. Observations that differ by more than this threshold are filtered out and the cubic spline is iteratively refit.
`weight`	When fitting the cubic splines, should individual observations be weighted by their year of acquisition relative to the focal year? If so, each observation is weighted by exp(-0.25*n.yrs.from.focal) when fitting the cubic splines.
`spl.fit.outfile`	(Optional) Name of output csv file containing the fitted cubic splines for each sample site. Useful for subsequent visualization.
`progress`	(TRUE/FALSE) Print a progress report?
`test.run`	(TRUE/FALSE) If TRUE, then algorithm is run using a small random subset of data and only a figure is output. This is used for model parameterization.

Value

Data.table that provides, for each observation, information on the phenological conditions for that specific day of year during the focal period. These data can then be used to estimate annual maximum spectral index and other growing season metrics using lsat_summarize_growing_season(). A figure is also generated that shows observation points and phenological curves for nine random sample locations.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt <- lsat_calc_spec_index(lsat.dt, 'ndvi')
# lsat.dt <- lsat_calibrate_rf(lsat.dt, band.or.si = 'ndvi', write.output = F)
lsat.pheno.dt <- lsat_fit_phenological_curves(lsat.dt, si = 'ndvi')
lsat.pheno.dt
```

`lsat_general_prep` *Prepare Landsat data for analysis*

Description

This function parses sample site coordinates and time period of each measurement, scales band values, and formats column names as needed for subsequent analysis using the lsatTS package.

Usage

```
lsat_general_prep(dt)
```

Arguments

`dt`	Data.table with Landsat data exported from Google Earth Engine using lsat_export_ts().

Value

Data.table with formatted and scaled values.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt
```

lsat_get_pixel_centers *Get Landsat 8 pixel centers for a polygon or a buffered point*

Description

A convenience helper function that determines the Landsat 8 grid (pixel) centers within a polygon plus an optional buffer. It can also be applied to a single point to retrieve all pixels within a buffer.

Does not work for large polygons. The default maximum number of pixels set by EE is 10000000 this should not be exceeded. Consider whether extraction for a large polygon is a good idea, if yes split the polygon into manageable chunks.

For the unlikely case that a polygon exceeds the boundaries of the Landsat tile closest to the polygon's center, the polygon is clipped at the boundaries of the Landsat tile and a warning is issued. Again, if this is the case, consider processing smaller polygons instead.

Please note: The approximation of the tile overlap with the polygon generates a warning by the sf package that the coordinates are assumed to be planar. This can be ignored.

Usage

```
lsat_get_pixel_centers(
  polygon_sf,
  pixel_prefix = "pixel",
  pixel_prefix_from = NULL,
  buffer = 15,
  plot_map = F,
  lsat_WRS2_scene_bounds = NULL
)
```

Arguments

polygon_sf	Simple feature with a simple feature collection of type "sfc_POLYGON" containing a single polygon geometry. Alternatively, a simple feature containing a simple feature collection of type 'sfc_POINT' with a single point.
pixel_prefix	Prefix for the generated pixel identifiers (output column "sample_id"). Defaults to "pixel".
pixel_prefix_from	Optional, a column name in the simple feature to specify the pixel_prefix. Overrides the "pixel_prefix" argument.
buffer	Buffer surrounding the geometry to be included. Specified in m. Defaults to 15 m - the nominal half-width of a Landsat pixel.
plot_map	Optional, default is FALSE. If TRUE the retrieved pixel centers and the polygon are plotted on a summer Landsat 8 image (grey-scale red band) using mapview. If a character is supplied an additional output to a file is generated (png, pdf, and jpg supported, see mapview::mapshot). Note: Both slow down the execution of this function notably, especially for large polygons! Only use in interactive R sessions.

`lsat_WRS2_scene_bounds`
: File path to the Landsat WRS2 path row scene boundaries. If not specified the boundaries are downloaded to a temporary file when the function is executed for the first time during a session. To avoid future downloads, the file may be downloaded manually and it's file path specified using this argument. The file can be found here: https://prd-wret.s3.us-west-2.amazonaws.com/assets/palladium/production/atoms/files/WRS-2_bound_world_0.kml See also: https://www.usgs.gov/core-science-systems/nli/landsat/
landsat-shapefiles-and-kml-files

Value

sf object of point geometries for Landsat 8 pixel centers within the polygon or the buffer around the point coordinate specified. For use in lsat_export_ts().

Author(s)

Jakob J. Assmann

Examples

```
# Using sf, dplyr, rgee and purr
library(sf)
library(dplyr)
library(rgee)
library(purrr)

# Initialize EE
ee_Initialize()

# Specify a region to retrieve pixel centers for
test_poly_sf <- list(matrix(c(-138.90125, 69.58413,
              -138.88988, 69.58358,
              -138.89147, 69.58095,
              -138.90298, 69.57986,
              -138.90125, 69.58413),
            ncol = 2, byrow = TRUE)) %>%
           st_polygon() %>%
           st_sfc(crs = 4326) %>%
           st_sf()

# Retrieve pixel centers and plot to mapview
pixels <- lsat_get_pixel_centers(test_poly_sf, plot_map = TRUE)


## Ge pixel centers for multiple regions
# Create multi-polygon sf
ellesmere <- st_polygon(list(matrix(c(-75.78526, 78.86973,
                                      -75.78526, 78.87246,
                                      -75.77116, 78.87246,
                                      -75.77116, 78.86973,
                                      -75.78526, 78.86973),
                                      ncol = 2, byrow = TRUE)))
zackenberg <- st_polygon(list(matrix(c(-20.56254, 74.47469,
                                    -20.56254, 74.47740,
                                    -20.55242, 74.47740,
```

```
                                    -20.55242, 74.47469,
                                    -20.56254, 74.47469),
                                  ncol = 2, byrow = TRUE)))
toolik <- st_polygon(list(matrix(c(-149.60686, 68.62364,
                                   -149.60686, 68.62644,
                                   -149.59918, 68.62644,
                                   -149.59918, 68.62364,
                                   -149.60686, 68.62364),
                                   ncol = 2, byrow = TRUE)))
test_regions_sf <- st_sfc(ellesmere, zackenberg, toolik, crs = 4326) %>%
  st_sf() %>%
  mutate(region = c("ellesmere", "zackenberg", "toolik"))

# Split and map lsat_get_pixel_centers using dplyr and purrr
pixel_list <- test_regions_sf %>%
   split(.$region) %>%
   map(lsat_get_pixel_centers,
       pixel_prefix_from = "region") %>%
   bind_rows()
```

`lsat_neighborhood_mean` *Compute Neighborhood Average Landsat Surface Reflectance*

Description

For each band, this function computes average surface reflectance across neighboring voxels at a sample site. Use this function when working with Landsat data extracted for buffered points. Also, make sure to have previously cleaning the individual observations using lsat_clean_data().

Usage

```
lsat_neighborhood_mean(dt)
```

Arguments

`dt`	A data.table containing coincident surface reflectance measurements for multiple Landsat pixels at each sample site.

Value

A data.table with average surface reflectance

`lsat_summarize_data_avail` *Summarize availability of Landsat data for each sample site*

Description

This little function summarizes the temporal period and availability of observations at each sample site.

Usage

```
lsat_summarize_data_avail(dt)
```

Arguments

`dt`	Data.table with columns named "sample.id" and "year".

Value

Data.table summarizing for each sample site the first, last, and number of years with observations, the minimum and maximum number of observations in a year, and the total number of observations across years. Also returns a figure showing the median (2.5 and 97.5 percentiles) number of observations per sample site across years for each Landsat satellite.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat_summarize_data_avail(lsat.dt)
```

`lsat_summarize_growing_seasons` *Summarize growing season characteristics using spectral vegetation indices*

Description

This function not only computes mean, median, and 90th percentile of a spectral index (SI) using observations for a user-specified "growing season," but also estimates the annual maximum SI and associated day of year using phenology modeling and growing season observations.

Usage

```
lsat_summarize_growing_seasons(
  dt,
  si,
  min.frac.of.max = 0.75,
  zscore.thresh = 3
)
```

Arguments

`dt`	Data.table generated by the function lsat_fit_phenological_curves().
`si`	Character string specifying the spectral vegetation index to summarize (e.g., NDVI).
`min.frac.of.max`	Numeric threshold (0-1) that defines the "growing season" as the seasonal window when the phenological curves indicate the SI is within a specified fraction of the maximum SI. In other words, an observation is considered to be from the "growing season" when the SI is within a user-specified fraction of the curve-fit growing season maximum SI.
`zscore.thresh`	Numeric threshold specifying the Z-score value beyond which individual observations are filtered before summarizing growing season SI.

Value

Data.table summarizing annual growing season conditions based on a spectral index.

Examples

```
data(lsat.example.dt)
lsat.dt <- lsat_general_prep(lsat.example.dt)
lsat.dt <- lsat_clean_data(lsat.dt)
lsat.dt <- lsat_calc_spec_index(lsat.dt, 'ndvi')
# lsat.dt <- lsat_calibrate_rf(lsat.dt, band.or.si = 'ndvi', write.output = F)
lsat.pheno.dt <- lsat_fit_phenological_curves(lsat.dt, si = 'ndvi')
lsat.gs.dt <- lsat_summarize_growing_seasons(lsat.pheno.dt, si = 'ndvi')
lsat.gs.dt
```