ISPRS Journal of Photogrammetry and Remote Sensing 144 (2018) 217–234

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

Towards a polyalgorithm for land use change detection

Rishu Saxena[a,*], Layne T. Watson[b], Randolph H. Wynne[c], Evan B. Brooks[c], Valerie A. Thomas[c], Yang Zhiqiang[d], Robert E. Kennedy[e]

[a] Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA
[b] Departments of Computer Science, Mathematics, and Aerospace and Ocean Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA
[c] Department of Forest Resources and Environmental Conservation, Blacksburg, VA 24061, USA
[d] Forest Ecosystems & Society, Oregon State University, Corvallis, OR 97331, USA
[e] College of Earth, Ocean, and Atmospheric Sciences, Oregon State University, Corvallis, OR 97331, USA

ARTICLE INFO

Keywords:
Time series analysis
Remote sensing
Change detection
Scalable computing
Polyalgorithm

ABSTRACT

One way of analyzing satellite images for land use and land cover change (LULCC) is time series analysis (TSA). Most of the many TSA based LULCC algorithms proposed in the remote sensing community perform well on datasets for which they were designed, but their performance on randomly chosen datasets from across the globe has not been studied. A polyalgorithm combines several basic algorithms, each meant to solve the same problem, producing a strategy that unites the strengths and circumvents the weaknesses of constituent algorithms. The foundation of the proposed TSA based 'polyalgorithm' for LULCC is three algorithms (BFAST, EWMACD, and LandTrendR), precisely described mathematically, and chosen to be fundamentally distinct from each other in design and in the phenomena they capture. Analysis of results representing success, failure, and parameter sensitivity for each algorithm is presented. For a given pixel, Hausdorff distance is used to compare the distance between the change times (breakpoints) obtained from two different algorithms. Timesync validation data, a dataset that is based on human interpretation of Landsat time series in concert with historical aerial photography, is used for validation. The polyalgorithm yields more accurate results than EWMACD and LandTrendR alone, but counterintuitively not better than BFAST alone. This nascent work will be directly useful in land use and land cover change studies, of interest to terrestrial science research, especially regarding anthropogenic impacts on the environment.

1. Introduction

Land use change is described as changes in how humans use the surface of the Earth (e.g., for agriculture, plantations, pastures, managed woods, conservation, settlements, or leaving it alone as natural ecosystem). Changes in land use lead to changes in albedo, thereby directly affecting the temperatures of the surrounding area. Significant and lasting changes in land use and land cover (LULC) have more profound effects. The past century has seen an exponential growth in human activities such as deforestation and urbanization causing significant changes in land cover in several parts of the world (Hansen et al., 2013). Simultaneously, significant changes in the global climate have also been observed, driven in part by LULC change (LULCC) (e.g., Fall et al., 2010). LULCC also has impacts on a wide variety of other ecosystem services. Monitoring LULCC across the globe, therefore, has become the need *du jour*. Land use change detection comprises any methodology used for determining the occurrence and nature of change in LULC.

Earth observation satellites (EOS) such as Landsat capture images of the Earth's surface at regular intervals using multiple spectral frequencies. These images hold valuable information that, if harnessed well, can be immensely helpful in understanding, monitoring, and managing our natural resources, as well as studying LULCC. One way of analyzing these satellite images for LULCC studies is time series analysis (or, temporal trajectory analysis). For time series analysis, several images of the scene under consideration, taken over a period of time, are stacked together chronologically and subsequently analyzed. Commonly, the time series for each pixel is treated individually; the full image stack is thus a collection of many time series. The choice of spectral band(s) varies from application to application. The objective is to discover a 'trend' in how different relevant variables (indicators) evolve over time. In change detection analysis, when the trajectory of one or more of the variables departs from the normal, a change is detected. Time series analysis for LULCC studies has been receiving increasing attention in the last decade, specifically, after the Landsat data

* Corresponding author.
E-mail address: rishus@vt.edu (R. Saxena).

https://doi.org/10.1016/j.isprsjprs.2018.07.002
Received 11 January 2018; Received in revised form 29 June 2018; Accepted 6 July 2018
Available online 27 July 2018

became freely accessible in 2008 (Woodcock et al., 2008). Several time series analysis algorithms have been proposed by different groups in the remote sensing community.

Despite a plethora of time series analysis algorithms available in remote sensing, design and selection of algorithms for LULCC detection in remote sensing appears to be almost always context specific. Most of the methods proposed to date seem to perform well on the type of data that they are designed for. Their performance on randomly picked datasets from across the globe has not been studied. The onus of choosing an appropriate algorithm that will perform well on their particular dataset falls on the user. Unfortunately, no single algorithm designed so far seems to work for all datasets (Cohen et al., 2017). For example, the Western Antarctica as well as the Greenland Ice Sheets are beginning to collapse due to global warming, the melting leading to continually receding snow covers at the respective locations. For these regions, using LULC algorithms based on periodicity assumptions is expected to lead to incorrect predictions and/or false alarms, although the nature and extent of this has not been studied yet. Even if there were no global warming, mild shifts in the 'phase' and 'amplitude' of seasons are known to take place (Petitjean et al., 2011). Time warping techniques (Petitjean et al., 2011) to deal with these issues may be helpful in some contexts, but their accuracy and scalability has not yet been satisfactorily investigated. Approaches based on periodicity and a moving window are possible, with additional computational costs.

A polyalgorithm is an effective strategy to unite the strengths and circumvent the weaknesses of multiple algorithms that are also individually designed to solve the same problem. The concept of polyalgorithm was introduced by Rice and Rosen (1966). A polyalgorithm uses a combination of several basic methods in a framework. Each of these basic methods is applicable to the same problem, with only their performance and/or success being different for different datasets (inputs). The construction of this framework involves experimenting with an increasingly heterogeneous set of situations to evolve a robust algorithm that is capable of choosing a correct subset of algorithms suitable for a given input, and has performance metrics to integrate their outputs. The details of algorithm selection and processing stay hidden from the user. Polyalgorithms have been designed in the past for solving various problems, for example, nonlinear systems of equations (Rice and Rosen, 1966, Rice, 1969, 2014, 1967), matrix computations on parallel architectures (Li, 1996, Häfner et al., 1999), and certain chemical models (Gomeni and Gomeni, 1979).

This work lays the foundation for a polyalgorithm for LULCC detection. Three currently existing, fundamentally different from each other, change detection algorithms are utilized. A similar work in this direction is Zhan et al. (2002), wherein a framework is developed to evaluate five different algorithms on the input dataset, compare them based on certain scores, and then return the best results. Similar approaches are also gaining ground recently in the field of classification algorithms (Dietterich et al., 2001, Kittler et al., 1998, Wozniak et al., 2014). Most recently, in Healey et al. (2017), multiple change detection algorithms are utilized to build a decision trees based ensemble algorithm for LULCC.

The rest of this paper is organized as follows: Section 2 presents background on state-of-the-art change detection algorithms available in remote sensing, puts them in the context of the general time series literature, and explains the choice of algorithms used in this work. Section 3 defines the notation. Sections 4–6 describe three different trend and change detection algorithms — EWMACD, BFAST, and LandTrendR; experimental results demonstrating the successes, failures, and sensitivity to parameters for each algorithm are presented. Prospects for a viable polyalgorithm are discussed in Sections 7 and 8 concludes with an assessment and future work.

2. Background

Most of the LULC algorithms proposed in the remote sensing literature can be divided into two categories: *bitemporal analysis* and *temporal trajectory analysis*. Bitemporal analysis was more popular before 2008 (when the availability of satellite data to the public was very limited) and forms the classical way of analyzing images — these algorithms analyze changes occurring between two images (dates). The more preferred bitemporal algorithms rely on image differencing (Banner and Lynham, 1981; Hame, 1986; Coppin and Bauer, 1994; Cohen and Fiorella, 1998; Cohen et al., 1998; Serneels et al., 2001), and linear transformations (Richards, 1984, Neilsen et al., 1998, Fung and LeDrew, 1987, Fung, 1990). Other strategies used to design bitemporal algorithms include image rationing (Jensen, 1983), image regression (Joyce and Burns, 1981), and composite analysis (Thomson et al., 1980). Detailed surveys of these algorithms can be found in Coppin et al. (2004), and Lu et al. (2004). Multi-Index Integrated Change Analysis (MIICA) (Jin et al., 2013), a recent popular algorithm, utilizes two Landsat image pairs and four different derived spectral indices for change detection. The interested reader is referred to Campbell and Wynne (2011) for a further decent categorization of these algorithms.

For time series analysis based change detection algorithms, the popular strategy has been to design pixel based algorithms, wherein the time series for one pixel at a time is analyzed. One strategy is to segment the time series into piecewise linear segments. Specifically, the time span is partitioned into intervals where each interval corresponds to a sustained trend in observed values. The boundaries of these intervals correspond to points of change or the start of a new trend. The number of intervals depends on how many changes in trends occurred for that time series. This approach is adopted, for example, in Cohen et al. (2010), Kennedy et al. (2007), de Jong et al. (2012, 2013), Verbesselt et al. (2010a,b). In Moisen et al. (2016), on the other hand, seven shapes that can possibly occur in time series spectral data are identified. Constrained regression is done using splines that can generate these shapes. Some methods leverage the fact that climate related phenomena such as vegetation, temperatures, and the like are expected to follow a periodic pattern and utilize models based on Fourier series (trigonometric polynomials) (Brooks et al., 2014, Zhu and Woodcock, 2014). In Vegetation Change Tracker (VCT) (Huang et al., 2010), another popular method, for *each* image, cloud, shadow, and water are first masked using histograms. A derived index based on the mean and standard deviation of observed values of multiple bands in that image is calculated. One (that with the best derived index) image per year is selected for further processing. Any masked values appearing in these selected images are filled in by interpolating two temporally nearest available values in the previous and subsequent years. Then a suite of decision rules is used to detect and classify forest disturbances. Wavelets were utilized in Cai and Desheng (2015).

Data mining approaches have been proposed for classification and change detection (Goodwin et al., 2008, Mougel and Folcher, 2012, Petitjean et al., 2012, 2010, Vintrou et al., 2012, 2013). In Goodwin et al. (2008), a decision tree classifier was used to detect an outbreak of mountain pine beetle. This algorithm was originally implemented only on a subset of all available Landsat images (specifically, one scence per year was chosen from a 14 year span). In Petitjean et al. (2010), sequential pattern mining was proposed for finding trends (and changes) in land cover; all images were utilized. Dynamic time warping (DTW) was proposed in Petitjean et al. (2011) for comparing and analyzing remote sensing time series as well as characterizing change. Other recent related references include Anees et al. (2016), Benedek et al. (2015) and Bouziani et al. (2010).

Improved trend approximation can be obtained if, instead of treating each pixel independently, information from nearby pixels (spatial information) is also utilized. One such approach is VeRDET (Hughes et al., 2017), which utilizes two-dimensional total variation regularization (TVR) (Rudin et al., 1992;Goldstein and Osher, 2009) to modify the images so that they have small-scale spatial patches (reduced spatial heterogeneity). TVR is then used again for fitting a piecewise linear polynomial to the time series of each pixel. In Petitjean

et al. (2012), each image in a stack is first segmented to generate region associated indices (in addition to the already existing spectral indices). Each pixel is thus characterized by both a spectral and a spatial index. Unsupervised classification algorithms are then used over this expanded set of indices for time series analysis and change detection. Other recent related references include Anees et al. (2016), Benedek et al. (2015), Bouziani et al. (2010), Chen et al. (2012a,b), Hussain et al. (2013), Iersel et al. (2018), Xiao et al. (2016), Xing et al. (2018), Gil-Yepes et al. (2016), and Zhe (2017).

While research on LULCC algorithms using satellite image time series is relatively new, development and use of time series analysis began nearly a century ago, with applications such as econometrics, seismology, weather prediction, electrocardiography, mathematical finance, control systems, and more. This has led to a plethora of algorithms for analysis and forecasting in the time series literature. Autoregressive moving average (ARMA) models, introduced by Wold (1938), are polynomial models for representing any stochastic process and making predictions (Whittle, 1951). For nonstationary data, the nonstationarity is eliminated by preprocessing the data. Specifically, the observed (input) values are replaced by the differences between their values and the previous values. The rest of the processing is carried out on this modified data using ARMA. This is known as the autoregressive integrated moving average (ARIMA) approach. The classic book by Box and Jenkins (1970) presented their method — a full framework for analyzing time series. This framework is polynomial based applying either ARMA or ARIMA models to find the best fit to a given time series. Approaches based on Fourier transforms (Agrawal et al., 1993), wavelets (Chan and Fu, 1999), support vector machines (Vlasveld, 2014), piecewise linear approximations (Hunter and McIntosh, 1999, Koski et al., 1995, Lavrenko et al., 2000, Keogh et al., 2001, and the references therein) are other currently popular methods for approximating time series. Regardless of the formalization, all the time series algorithms in the literature fit in one of the following classes:

(i) Kernel regression methods. These methods represent the time series as a linear combination of basis functions. Typically, a linear system of equations is solved to determine the coefficients. Any analysis and predictions are done based on this representation. ARMA models (polynomial regression), Box-Jenkins models (polynomial regression), Fourier transform based approaches (trigonometric polynomials), and wavelet transforms would all classify as kernel regression.
(ii) Top-down approaches. In these algorithms, an approximation is first made to the whole time series. Then, typically using error estimate criteria, finer partitions of the time series are sought so that each new partition is a refinement of the previous partition. This is repeated until either a maximum number of iterations is reached or each segment of the partition satisfies a convergence criterion. The 'Iterative End Points Fits' algorithm (Ramer, 1972) is an example of a top-down approach. Other top-down algorithms include Douglas and Peucker (1973), Duda and Hart (1973), Li et al. (1998), Shatkay and Zdonik (1996).
(iii) Bottom-up approaches. Bottom-up approaches represent the most elementary units of the data first; on each iteration, increasingly larger (or coarser or more 'complex') structures are composed from the simpler structures and their evaluations in the previous iteration. Dynamic programming and all recursive algorithms classify as bottom-up algorithms, which are frequently implemented using backtracking.

A list of some recent change detection algorithms in remote sensing and their classification is displayed in Table 1.

A strategy to construct a polyalgorithm would be to include algorithms that are fundamentally distinct from each other in terms of the phenomenon they capture as well as in construction. However, the suitability of available algorithms to capture different scenarios is currently not fully known. Therefore, choosing an algorithm from each of the classes in Table 1, ensuring some variation in type of phenomenon captured, may be used as a first step towards polyalgorithm construction. In this paper, LandTrendR, EWMACD, and BFAST are studied. LandTrendR, a top-down approach, generates a piecewise linear model to represent the input time series. A broad trend can therefore be captured. It is not expected to be sensitive to seasonal deviations/anomalies, though. For a stationary time series whose periodicity or variation changes, the performance of LandTrendR is yet unknown. EWMACD is a kernel regression method based on harmonic regression, designed to detect any persistent deviations from the stable pattern observed during a training period chosen by the user. This algorithm performs very well in regions where the land cover exhibits strong periodicity. However, its performance on time series with aperiodic variations and/or unstable training periods is limited. BFAST, a bottom-up approach, is more generic and models both linear trends and seasonal variations. BFAST's periodic linear model is more accurate than that of LandTrendR, since BFAST recursively evaluates the possibility of every single time point being a breakpoint, and then chooses the most optimal set of breakpoints. Unfortunately, such an exhaustive search makes BFAST more expensive than other algorithms (Saxena et al., 2017a). Also, the correctness of BFAST seasonal fits is yet to be tested. Finally, the current algorithms were originally designed for forest covers and have only been tested on the same so far; their performance on nonforest land covers (e.g., arid areas) has not been studied.

Table 1
General time series literature classification of some remote sensing algorithms.

	Algorithms in remote sensing	Segmentation approaches in general time series literature
(i)	CCDC, EWMACD, SHAPE-SELECT-FOREST	Kernel regression method
(ii)	LandTrendR, VeRDET	Top-down approach
(iii)	BFAST, MIICA, VCT	Bottom-up approach

Sections 4–6 elaborate on each of these three algorithms and present instances of their success and failure.

3. Preliminaries

Notation and definitions. For an $m \times n$ matrix A, an n-vector x, $I \subset \{1, \ldots, m\}$, $J \subset \{1, \ldots, n\}$, let A_{IJ} denote the submatrix of A formed from the rows indexed by I and the columns indexed by J, and x_J denote the subvector of x indexed by J. $A_{I\cdot}$ ($A_{\cdot J}$) are the rows (columns) of A indexed by I (J), respectively. An image is an $R \times C$ matrix D, where each D_{rc} (pixel) is an $S \times B$ matrix, whose (s, b) element $(D_{rc})_{sb}$ is the signal value at time index s and frequency band index b. S' is the number of missing data values and $\overline{S}$ is the total number of timestamps provided in the data, i.e., $\overline{S} = S + S'$.

Input data. Images taken by satellites Landsat 7 and 8 are used for the experiments presented in this paper. Landsat 7 is equipped with an enhanced thematic mapper (ETM+) instrument, which takes images at 30 m resolution, using eight different spectral bands (one image per band). Landsat 8 employs an operational land imager (OLI) for the same task. Any given area of the Earth's surface is captured every 16 days. The experiments presented in this paper utilize normalized difference vegetation index (Tucker, 1979, Kriegler et al., 1969)

$$\mathrm{NDVI} = \frac{\mathrm{NIR}-\mathrm{R}}{\mathrm{NIR} + \mathrm{R}},$$

where NIR is the near infrared band (band 4) and R is the visual red band (band 3). In the remote sensing community, NDVI is widely considered to be a good metric for identifying the presence of vegetation cover. NDVI is directly related to the photosynthetic capacity of

plant canopies. Broadly, forests typically tend to have high positive values (e.g., 0.6–0.8), scrubs/shrubs lean towards slightly smaller values, and any other land covers with smaller canopy cover (e.g., meadows, grazing areas) have even lesser values. Persistently decreasing NDVI values usually indicate decreasing foliage — which could be because of harvest (steep negative slope), insects (gentler negative slope), seasons (descending portions of periodic curves), or any other reason. Persistently increasing NDVI values indicate leaf cover increasing. For the experiments presented here, negative values of NDVI are deemed irrelevant as they correspond to water, clouds, or missing observations, and are masked out.

Reference data. TimeSync data, a dataset that is based on human interpretation of Landsat time series, is used for validating the results for experiments in 1D. This dataset is prepared using TimeSync Landsat images visualization and change data collection tool (Cohen et al., 2010). This tool enables disturbance characterizations for pixel-level samples of Landsat time series data, relying on human interpretations of change as viewed in image chip series, spectral index trajectories, high spatial resolution image temporal snapshots from Google Earth, and other supporting products. The current dataset was built using Landsat image stacks belonging to six different path/rows (scenes) and spanning the years 1984 to 2014. From each scene, 300 pixels were chosen with random sampling, and without regard to land cover. There are 1800 change pixels in the dataset. For each of these pixels, the following attributes were noted: occurrence of disturbance, the first year of detection (a year between 1986 and 2011), the duration for gradual disturbances (in number of years), and the causal agent class (harvest, fire, mechanical, decline, wind, other). Similar to the approach adopted in Brooks et al. (2017), all data marked by the associated Fmask codes (Zhu and Woodcock, 2012) as nonclear was excluded, and then the NDVI values were calculated.

Evaluation. The three algorithms presented in this paper are applied to NDVI data. The TimeSync dataset is used as the reference data set. Due to a focus on change detection, only experience with change pixels is discussed in this paper. The time period of 2000–12 is utilized. Success/failure of each algorithm is compared to TimeSync data by checking the disturbance year(s) stated in the dataset with the year(s) predicted in the outcome of respective algorithms. Note that TimeSync itself is an interpreted dataset, not ground truth. For the pixels explicitly displayed in this paper, visual comparison of outcome with trajectory is also done. Published values of parameters are used. Sensitivity of algorithms to parameters is also presented. In Section 7, mathematical distances (Hausdorff distances) are utilized to further provide quantify the differences in outcomes of different algorithms.

The three algorithms presented in this paper are applied to NDVI data. TimeSync dataset is used as reference data set. Due to focus on change detection, only experience with change pixels is discussed in this paper. The time period of 2000–12 is utilized. Success/failure of each algorithm is compared to TimeSync data by checking the disturbance year(s) stated in the dataset with the year(s) predicted in the outcome of respective algorithms. Note that TimeSync itself is an interpreted dataset, not ground truth. For the pixels explicitly displayed in this paper, visual comparison of outcome with trajectory is also done. Published values of parameters are used. Sensitivity of algorithms to parameters is also presented. In Section 7, mathematical distances (Hausdorff distances) are utilized to further provide quantify the differences in outcomes of different algorithms.

4. Exponentially weighted moving average change detection (EWMACD)

EWMACD, proposed in Brooks et al. (2014), is a kernel regression approach modeling the time series as a linear combination of trigonometric polynomials. The model is trained over data collected in the initial two (or more) years. When the observations in the subsequent years deviate from the values forecast by the model for a 'substantial' length of time (persistence), a change is declared (recovery or disturbance). The training period as well as the persistence are parameters of the algorithm. A positive flag is raised in instances of sustained growth while sustained losses are indicated by negative flags. EWMACD is able to capture seasonal changes, but is also very sensitive to several algorithm parameters.

Algorithm EWMACD.

for row r 1 **step** 1 **until** R
for column c 1 **step** 1 **until** C **do**

Step 1: Write the time series data in the column $(D_{rc})_{\cdot b}$ as

$$(D_{rc})_{\cdot b} = \begin{pmatrix} u \\ v \end{pmatrix},$$

where the M-dimensional vector u is deemed training data and the $(S-M)$-dimensional vector v as the test data. Let

$$X = \begin{bmatrix} 1 & \sin t_1 & \cos t_1 & \cdots & \sin Kt_1 & \cos Kt_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \sin t_M & \cos t_M & \cdots & \sin Kt_M & \cos Kt_M \end{bmatrix}$$

be the Gram matrix for the time points $t_1, \ldots, t_M$, using K harmonics, where $M > 2K + 1$. The least squares fit to the training data u is then written as

$$u(t) = \alpha_0 + \sum_{i=1}^{K} (\alpha_{2i-1} \sin it + \alpha_{2i} \cos it)$$

with coefficients

$$\alpha = (X^t X)^{-1} X^t u$$

and residual

$$E(\alpha) = u - X\alpha.$$

Remark 1. In practice α is computed via a QR factorization of X, not by computing $(X^t X)^{-1}$ explicitly.

Next let $I = \{i | |E(\alpha)_i| < \gamma_1\}$, where γ_1 is a user defined threshold and $|I| > 2K + 1$. Calculate the coefficients for an improved fit to the underlying signal as

$$\alpha^* = ((X_{I\cdot})^t X_{I\cdot})^{-1} (X_{I\cdot})^t u_I.$$

With the refined coefficients α^*, calculate the residuals for

(i) the complete time series $(D_{rc})_{\cdot b}$ as

$$E^*(\alpha^*) = (D_{rc})_{\cdot b} - \overline{X}\alpha^*,$$

where $\overline{X}_{s\cdot} = (1, \sin t_s, \cos t_s, \ldots, \sin Kt_s, \cos Kt_s)$, for $s = 1, \ldots, S$.

(ii) the outlier-free time series as $(E^*(\alpha^*))_{\overline{I}}$, where $\overline{I} = \{s | |E^*(\alpha^*)_s| < \gamma_2\}$, γ_2 is a user defined threshold, and

(iii) the outlier-free training set $\hat{I} = \overline{I} \cap \{1, \ldots, M\}$ as

$$(E^*(\alpha^*))_{\hat{I}} = u_{\hat{I}} - X_{\hat{I}\cdot}\alpha^*,$$

where $|\hat{I}| > 2K + 1$.

Remark 2. In the present implementation,

$$\gamma_2 = \begin{cases} 1.5\eta, & i \in [1, M], \\ 20\eta, & i \in (M, S], \end{cases}$$

where η is the standard deviation of the first M elements of the residual vector $E^*(\alpha^*)$.

Step 2: Define the control limit vector τ by

$$\tau_i = \mu + \sigma L\sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2i})},$$

for $i = 1, 2, ..., |\bar{I}|$, where $\mu = 0$ is used here, σ is the standard deviation of the outlier-free training data errors $(E^*(\alpha^*))_{\hat{I}}$, L is the multiple of this standard deviation σ, and $\lambda \in (0, 1]$ is the weight given to the most recent residual in the exponentially weighted moving average (EWMA) defined next. L is typically set to 3 or slightly smaller depending on the value of λ.

Step 3: Let $\bar{I} = \{j_1, j_2, ..., j_{|\bar{I}|}\}$, $j_1 < j_2 < \cdots < j_{|\bar{I}|}$. Define the vector z by

$$z_1 = (E^*(\alpha^*))_{j_1},$$
$$z_i = (1-\lambda)z_{i-1} + \lambda(E^*(\alpha^*))_{j_i}, \quad i = 2, ..., |\bar{I}|.$$

This is the exponentially weighted moving average (EWMA) of the residual $(E^*(\alpha^*))_{\bar{I}}$.

Step 4: Define the flag history S-vector f by

$$f_s = \begin{cases} \text{sgn}(z_i)\lfloor |z_i/\tau_i| \rfloor, & s = j_i \in \bar{I}, \\ 0, & \text{otherwise.} \end{cases}$$

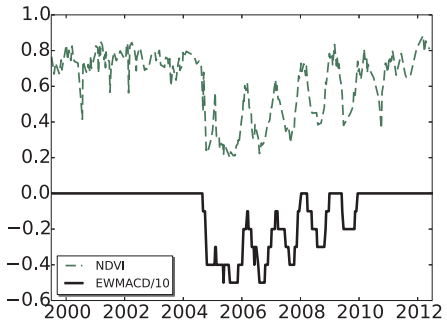
If there is a run of $+1$ or -1 in the values $\text{sgn}(\Delta f_s) = \text{sgn}(f_{s+1} - f_s)$ of length ϖ, called the 'persistence', signal a change at the index s beginning the (nonzero) run.

Remark 3. Missing data is automatically handled by not assuming that the time points t_i are equally spaced. Alternatively, missing data for time point t_k can be handled by including t_k in the sequence $(t_1, t_2, ..., t_S)$, but excluding t_k from the training sequence $(t_1, t_2, ..., t_M)$ and k from the sets I, $\bar{I}$, and $\hat{I}$, which is equivalent to treating $(D_{rc})_{kb}$ as an outlier and to setting the flag $f_k = 0$.

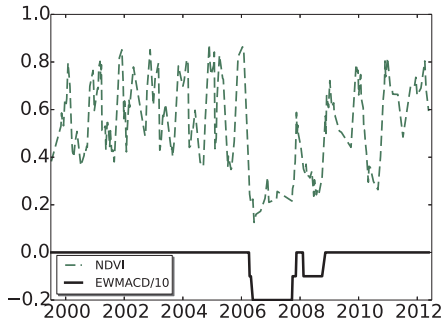
end
end

The complexity of this approach is $\mathcal{O}((2K+1)^2 S)$. EWMACD outcomes are expected to be zeros when the time series trajectory matches the trajectory in the training period. Nonzeros are expected to appear when the trajectory deviates from the training period for a substantial length of time (determined by control chart parameters and the persistence). The first timepoint when the outcome trajectory deviates from the prior constant (or zero) value is the estimated date of disturbance, the subsequent nonzero sequence of values provides recovery (or, loss) period information. A sharp loss (e.g., harvest) is expected to manifest as a sudden drop in the outcome as well. A gentler loss (e.g., mountain pine beetle) is expected to appear in the form of the outcome trajectory gradually drifting away from zero (with an overall negative slope). Similarly, recovery will appear in the form of the trajectory following an overall positive slope, revealing recovery period information. In Brooks et al. (2017), Edyn, an improved version of EWMACD that retrains data after a disturbance is sensed, is proposed (this development not implemented here). In any case, since the training data in the postshock period is unstable, the training period of Edyn after the first break is expected to be unstable. For all the experiments presented in this work, the first two years of data is used for training EWMACD. The rest of the parameter values for EWMACD are as follows: $K = 2$, $L = 0.5$, $\lambda = 0.3$, $\varpi = 7$.

Success. Fig. 1 displays the outcome of EWMACD on NDVI data for two pixels. Fig. 1(a) has a stable trajectory with high NDVI values until 2004. The NDVI values drop suddenly in 2004 from 0.8 to 0.2, indicating the possibility of an event. Gradual recovery is seen after that. Per TimeSync data, this is a forest pixel where harvest occurred in 2004. The trajectory and the TimeSync information are, thus, in agreement for this pixel. Using the published parameters, EWMACD



(a) Forest, Harvest in 2004



(b) Agriculture → NVA, Clearing in 2006

Fig. 1. EWMACD success. Flag history (divided by 10) signals times of change by runs of increases or decreases in the flags. (a) Harvest in a forest; (b) Fire in a nonforest pixel.

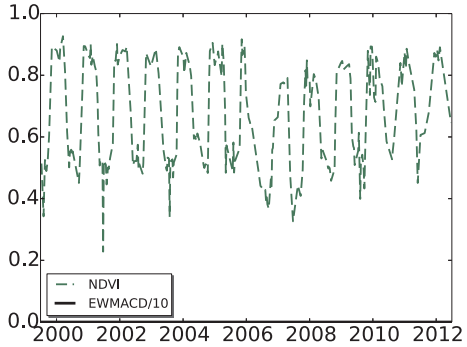
captures the occurrence and timing of the harvest accurately. The nonzero portion of the curve following this until 2010 displays the recovery period information, which exhibits seasonal effects, likely related to the relatively flat training curve.

In Fig. 1(b), a stable trajectory with mean 0.6 and much variance is seen until 2006. In 2006, the values drop to 0.2 (perhaps very little leaf cover left). Subsequently, some leaf cover is regained but the trajectory is different from the pre-2006 trajectory. TimeSync data states that this pixel is used for agriculture initially, is cleared in 2006–07, and used for nonvegetated anthropogenic purposes after that. The trajectory and the TimeSync data are in agreement. EWMACD accurately detects the sudden absence of vegetation cover. Gradual increase in NDVI is indicated thereafter.

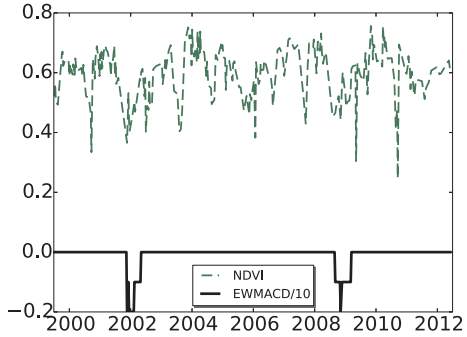
Failure. Failure of EWMACD when the training period is not stable is trivial. Instances of failure when the training period is stable are discussed here. Fig. 2(a) displays a forest pixel. The NDVI trajectory shows a disturbance in 2006. TimeSync data states a harvest in 2005–06. EWMACD misses this event. The outcome does not change with different parameters either, possibly because the change is relatively brief compared to the training data.

In Fig. 2(b), the trajectory appears stable except for two visible drops in 2002 and 2008. TimeSync data assigns nonvegetated anthropogenic (NVA) land cover to this pixel, and no recorded LULCC event in the 2000–12 time period. EWMACD shows two instances of loss, which seem to agree with the trajectory but not so with TimeSync. The second signal (in 2008–09) vanishes when three harmonics are used instead of two, indicating that this trajectory could be following time periods that are different from those used in EWMACD. The first signal, however, remains through a range of parameters, and could potentially be attributed to an insufficient training period.

Sensitivity. The accuracy of EWMACD is expected to be influenced by the parameters L and λ used for the control limits calculation (Step 2 of the algorithm), the persistence parameter (ϖ), and the training period. Sensitivity to λ is presented here. The forest pixel displayed in

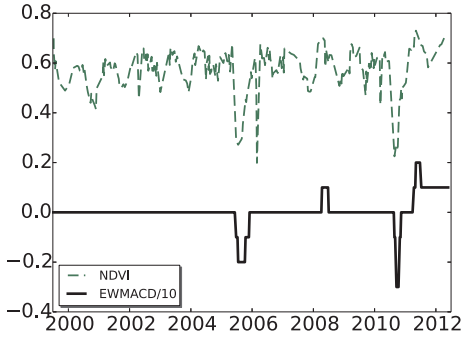


(a) Forest, Harvest in 2005–06

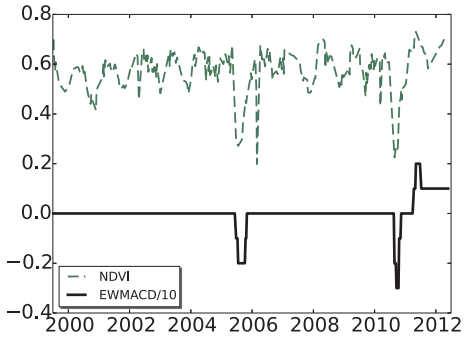


(b) NVA, No recorded change

Fig. 2. EWMACD failure. (a) Misses a harvest in a forest; (b) false alarms in a developed area.



(a) $\lambda = 0.5$



(b) $\lambda = 0.58$

Fig. 3. EWMACD sensitivity. A false alarm appears for $\lambda = 0.5$, but not for $\lambda = 0.58$. For $\lambda = 0.6$, the second event also disappears.

Fig. 3 is where a fire is known to have occurred in 2005–06. The NDVI trajectory shows a drop in 2005–06. For $\lambda = 0.5$ (Fig. 3(a)), EWMACD detects the fire in 2005–06. In addition, it shows a second flag in 2008, and a third flag in 2010–11 followed by recovery. The second flag is likely a false alarm. The third flag corresponds to a visible dip (potentially, an event not recorded) in NDVI at that time. The false alarm does not appear for $\lambda = 0.58$ (Fig. 3(b)). On using $\lambda = 0.6$, the flag in 2010–11 also disappears (which may make the outcome agree with TimeSync data but not with the trajectory). Using $\lambda = 0.4$, on the other hand, produces more false alarms between 2000 and 2006.

In general, decreasing L and ϖ has the effect of increasing the sensitivity of the algorithm. This means that the number of false alarms may also increase. λ determines the restrospectiveness of the algorithm. Decreasing λ should mask abrupt changes (the kind displayed in examples here) in favor of highlighting subtle, chronic changes (the kind TimeSync might also pick up). Finally, since this algorithm utilizes a training period, it is important that there be no change during the training period. When the training period does happen to contain a change, the signals' signs are expected to skew in the direction opposite to the direction of 'the change during the training period', but this needs to be investigated further. In Edyn (Brooks et al., 2017), an improved version of EWMACD, the harmonic model coefficients and the persistence are dynamically updated. Specifically, when EWMACD finds an event, the following (two) years are used to retrain the model, and the updated model is used for sensing change thereafter. This may alleviate the postchange false alarms to some extent.

5. LandTrendR

LandTrendR was proposed in (Kennedy et al., 2010). Starting with a linear regression fit to the entire time series, LandTrendR partitions the time series step by step, adding breakpoints (called vertices here) at each step. A set of potential vertices is thus generated in a straight top-down approach. Once these vertices have been generated, they are refined in multiple passes: (i) first the least influential vertices (corresponding to most obtuse angles) are discarded; (ii) then, of the now remaining vertices, each vertex is dropped one by one, based on a continuous piecewise linear fit (a 'model') using anchored regression (anchored regression comprises doing a least squares fit to the data but with one end fixed and only the slope (or the second end) to be determined); and (iii) the goodness of each model is evaluated in terms of F-statistics ('improvement compared to the *mean* model'). Of all the potential models, the model with lowest p value of the F-statistic is chosen as the final model[1]. (also called intercept-only model, intercept being equal to the average of the observations under consideration.) LandTrendR's original implementation utilized only one image per year from the Landsat image stacks, out of the 23 or so available per year. However, in this work, all available images are utilized.

Algorithm LandTrendR.

for row r 1 **step** 1 **until** R **do**
for column c 1 **step** 1 **until** C **do**

Step 1: Despike Let $u = (D^0_{rc})_{,b}$ denote the raw time series data. For each time point t_i, $1 < i < S$, define

$\Delta u_i = (D^0_{rc})_{(i+1)b} - (D^0_{rc})_{ib}$, $\nabla u_i = (D^0_{rc})_{ib} - (D^0_{rc})_{(i-1)b}$, $\mu\delta u_i$, and
$= (D^0_{rc})_{(i+1)b} - (D^0_{rc})_{(i-1)b}$, $k_i = 1 - |\mu\delta u_i| / \max\{|\nabla u_i|, |\Delta u_i|\}$

correction $\kappa_i = (\delta^2 u_i) k_i/2 = ((D^0_{rc})_{(i-1)b} - 2(D^0_{rc})_{ib} + (D^0_{rc})_{(i+1)}) k_i/2$.

For each i such that $k_i = \max_{1<j<S} k_j$, update $(D_{rc})_{ib} (D^0_{rc})_{ib} + \kappa_i$. Repeat iteratively until $\max_{1<j<S} k_j < \upsilon$, some given despiking tolerance.

Step 2: Find potential breakpoints Let $S^1 = (t_1, \ldots, t_S)$ be the original sequence of time points and $I^1 = (2, \ldots, S-1)$ denote the corresponding sequence of interior indices. Let

$$X = \begin{pmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_S \end{pmatrix}$$

be the Gram matrix for the time points $t_1, \ldots, t_S$, for ordinary least squares linear regression. The least squares fit to this data is given by

$$u(t) = \alpha_0 + \alpha_1 t$$

with coefficients

$$\alpha = (X^t X)^{-1} X^t u$$

and residuals

$$E^1(\alpha) = u - X\alpha.$$

Find the smallest index i_1 corresponding to the maximum absolute deviation, i.e.,

$$i_1 = \min\{i | i \in I^1 \quad \text{and} \quad |E^1(\alpha)_i| = \max_{j \in I^1} |E^1(\alpha)_j|\}.$$

Split the sequence S^1 into two subsequences,

$$S_l^1 = (t_1, \ldots, t_{i_1}) \text{and} S_r^1 = (t_{i_1}, \ldots, t_S).$$

Do linear regression on each of these and compute their respective mean squared errors, MSE_l and MSE_r. Suppose $|\text{MSE}_l| \leqslant |\text{MSE}_r|$. Then let $S^2 = S_r^1$ with interior index set $I^2 = (i_1 + 1, \ldots, S-1)$ be the next candidate sequence for 'breakpoint search'.

Again, find the smallest index i_2 corresponding to the maximum absolute deviation

$$i_2 = \min\{i | i \in I^2 \text{and} |E^2(\alpha^2)_i| = \max_{j \in I^2} |E^2(\alpha^2)_j|\}.$$

Again, split S^2 into two subsequences $S_l^2 = (t_{i_1}, \ldots, t_{i_2})$ and $S_r^2 = (t_{i_2}, \ldots, t_S)$, compute the least squares fit for each of these, choose the interval with higher MSE, and find the index i_3 corresponding to maximum absolute deviation. Recursively apply the algorithm until there are $\mu + \nu + 1$ breakpoints (including t_1 and t_S), where μ is the maximum number of segments allowed and ν is the maximum number of vertex overshoots (see Step 3 below) allowed. (In the rare circumstance that $\text{MSE}_l = \text{MSE}_r = 0$ at some iteration, there may be fewer than $\mu + \nu + 1$ breakpoints.)

Let $\overline{S} = (t_1, t_{i_1}, \ldots, t_{i_{\mu+\nu-1}}, t_S)$ be the final sequence of (sorted) breakpoints thus obtained, $\overline{I} = (1, i_1, \ldots, i_{\mu+\nu-1}, S)$ and $\overline{V} = (v_1, v_{i_1}, \ldots, v_S)$ be the corresponding index and 'vertex' sequences, where $v_i = (t_i, u_i)$.

Step 3: Cull by angle change

Define the sequence of angles

$$\alpha_j = \arccos\left(\frac{(v_{\overline{I}_j} - v_{\overline{I}_{j-1}}) \cdot (v_{\overline{I}_{j+1}} - v_{\overline{I}_j})}{\|(v_{\overline{I}_j} - v_{\overline{I}_{j-1}})\| \|(v_{\overline{I}_{j+1}} - v_{\overline{I}_j})\|} \right), \quad \text{for } j = 2, 3, \ldots, \mu + \nu.$$

Find $\overline{a} = \min\{i | \alpha_i = \min_j \alpha_j\}$, delete $v_{\overline{a}}$ from the sequence $\overline{V}$, and recalculate from this the angles with the new vertices. Repeat until reaching the sequence $V^* = (v_1, v_{l_1}, \ldots, v_{l_{\mu-1}}, v_S)$ with index sequence $L^* = (1, l_1, \ldots, l_{\mu-1}, S)$.

Step 4: Fit trajectories

Moving from $i = 1, \ldots, \mu$, consider consecutive vertices $v_{L_i^*}, v_{L_{i+1}^*} \in V^*$, one at a time, and an anchored regression fit

$$u_{AR}(t) = y_{L_i^*} + \alpha(t - t_{L_i^*}),$$

where $y_{L_i^*}$ is the 'fitted' value inferred from the fit in the preceding interval $(t_{L_{i-1}^*}, t_{L_i^*})$, α is the solution to the least squares regression problem $u_{AR} \approx u$ at the points $t_{L_i^*+1}, \ldots, t_{L_{i+1}^*}$. For the special case $i = 1$, the coefficient $y_{L_1^*}$ is also estimated. The final result of this step will be a continuous piecewise linear function $P^*(t)$ covering the full domain. Further, let $Y^* = (y_1, y_{l_1}, \ldots, y_{l_{\mu-1}}, y_S)$ be the sequence of fitted values at the breakpoints with indices L^*. Call the tuple $\mathscr{M}^* = (P^*(t), L^*, V^*)$ a regression model. In addition, let $(y_1, y_2, \ldots, y_S)$ be the sequence of fitted values over all time points in S^1 as predicted by $\mathscr{M}^*$.

Step 5: Model statistics

The improvement in prediction from regression compared to the mean model is given by the random variable

$$X_1^2 = \sum_{i=1}^{S} (u_i - \overline{u})^2 - \sum_{i=1}^{S} (u_i - y_i)^2 = \sum_{i=1}^{S} (y_i - \overline{u})^2,$$

where $\overline{u}$ is the mean value of the observations. The squared distance of the observed values from the values predicted by the regression model is the random variable

$$X_2^2 = \sum_{i=1}^{S} (u_i - y_i)^2.$$

Assuming that $y_i - \overline{u}$ and $u_i - y_i$ are independent, normally distributed, and have variance one, X_1^2 and X_2^2 have a χ^2 distribution with degrees of freedom $d_1 = \mu$, $d_2 = S - \mu - 1$, respectively. Therefore, the ratio

$$F = \frac{X_1^2/d_1}{X_2^2/d_2}$$

has an F-distribution and F-statistics can be used for measuring the 'goodness' of fit of the regression model. Let f be the F-statistic for model $\mathscr{M}$. Calculate the p-value of this F-statistic:

$$\begin{aligned} Q(f|d_1, d_2) &= 1 - I_{\frac{d_1 f}{d_1 f + d_2}}\left(\frac{d_1}{2}, \frac{d_2}{2}\right) \\ &= I_{\frac{d_2}{d_2 + d_1 f}}\left(\frac{d_2}{2}, \frac{d_1}{2}\right). \end{aligned}$$

$Q(f|d_1, d_2)$ is the probability that $F > f$ and $I_x(a, b)$ denotes the regularized incomplete Beta function given by

$$I_x(a, b) = \frac{1}{B(a, b)} \int_0^x t^{a-1}(1-t)^{b-1} dt,$$

$a > 0$, $b > 0$, where

$$B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a + b)} = \int_0^1 t^{a-1}(1-t)^{b-1} dt$$

is the (complete) Beta function, and

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt, \qquad \Re z > 0$$

is the Gamma function.

For model $\mathscr{M}^{(\mu)} = \mathscr{M}^*$, where the superscript corresponds to the number of segments in the model, let the p-value calculated in this step be $p^{(\mu)}$.

Step 6: Generate more (simpler) models

Begin with the model $\mathscr{M}^{(\mu)} = (P^{(\mu)}, L^{(\mu)}, V^{(\mu)}) = (P^*(t), L^*, V^*)$.

(i) Assume that a disturbance always corresponds to a positive slope while a negative slope indicates recovery.

First look for negative slopes at interior vertices. Let $V_{i^{(\mu)}}^{(\mu)} \neq V_1^{(\mu)}$ be the left vertex (leftmost in case of a tie) of the segment with steepest negative slope. Delete $L_{i^{(\mu)}}^{(\mu)}$ from the index sequence $L^{(\mu)}$, giving the shorter sequence $L^{(\mu-1)}$.

However, if no negative slopes are found or, if the steepest negative segment happens to be the leftmost segment of the current model, then for each interior vertex $v_{L_i}^{(\mu)}$ in the model, consider the point to point connect using the vertices immediately to the left and right of $v_{L_i}^{(\mu)}$, i.e., $v_{L_{i-1}}^{(\mu)}$ and $v_{L_{i+1}^{(\mu)}}$:

$$u_{PP}^{(\mu)}(t) = \frac{y_{L_{i+1}^{(\mu)}} - y_{L_{i-1}^{(\mu)}}}{t_{L_{i+1}^{(\mu)}} - t_{L_i^{(\mu)}}}(t - t_{L_{i-1}^{(\mu)}}) + y_{L_{i-1}^{(\mu)}},$$

for $i \in L^{(\mu)} \setminus \{1, S\}$, and calculate the $\mathrm{MSE}_{L_i^{(\mu)}}$ for vertex $v_{L_i^{(\mu)}}$ as

$$\mathrm{MSE}_{L_i^{(\mu)}} = \frac{1}{t_{L_{i+1}^{(\mu)}} - t_{L_{i-1}^{(\mu)}}} \sum_{k=L_{i-1}^{(\mu)}}^{L_{i+1}^{(\mu)}} (u_{PP}^{(\mu)}(t_k) - u_k)^2,$$

for $i \in L^{(\mu)} \setminus \{1, S\}$. Then define $i^{(\mu)} = \min\{i | \mathrm{MSE}_{L_i^*} = \min_j \mathrm{MSE}_{L_j^*}\}$, the index of the vertex dropping which leads to least MSE. Delete $L_{i^{(\mu)}}^{(\mu)}$, resulting in the shorter sequence $L^{(\mu-1)}$.

(iii) Remove the corresponding vertex from $V^{(\mu)}$ giving $V^{(\mu-1)}$, and as in Step 4 generate the new piecewise linear fit $P^{(\mu-1)}(t)$ and model $\mathscr{M}^{(\mu-1)} = (P^{(\mu-1)}(t), L^{(\mu-1)}, V^{(\mu-1)})$.

(iii) Calculate the p-value $p^{(\mu-1)}$ for this model.

Proceeding in this way, generate a total of μ models $\mathscr{M}^{(i)}$, $i = \mu, \ldots, 1$.

Step 7: Pick best model $\mathscr{M}_{i^*}$.

Let i^* be the smallest index i corresponding to the models $\mathscr{M}^{(i)}$ whose p-value is less than a user defined recovery threshold τ, i.e., $i^* = \min\{i | p^{(i)} \leqslant \tau, i = 1, \ldots, \mu\}$.

Remark. Check the linear segment slopes. If, for any model $\mathscr{M}^{(i)}$ under consideration, the recovery (to a global baseline) happens quicker than the quickest disturbance ϱ (from a global baseline), that model is discarded.

Step 8: Alternate approach.

If no models are found using Steps 1–7, repeat Steps 4–7 with the following modifications:

Step 4′. Instead of computing the continuous piecewise linear approximation $P^*(t)$ one segment at a time, going from left to right, compute $P^*(t)$ using all the data at once. This is done by expressing $P^*(t)$ as a linear combination of B-splines of order 2 with knot sequence $(t_1, t_1, t_{l_1}, t_{l_2}, \ldots, t_{l_{\mu-1}}, t_S, t_S)$, and then solving a linear least squares problem for the $\mu + 1$ coefficients of these B-spline basis functions. (Note that there is no need to use the Levenberg–Marquardt algorithm as proposed in Kennedy, Yang, and Cohen (2010)).

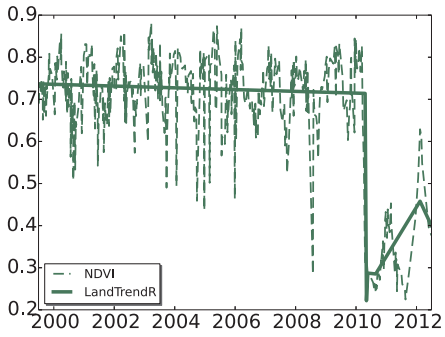
Step 6′. Skip directly to the point to point connect approach, without looking for negative slopes at all.

end

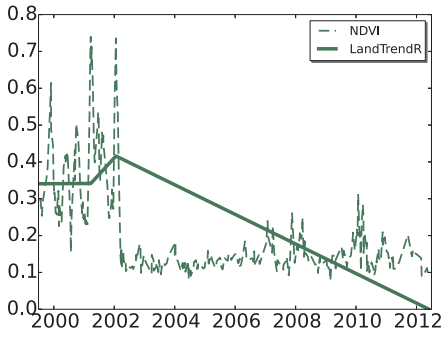
end

The complexity of the algorithm is $n_b \mathscr{O}(S)$, where $n_b = \mu + \nu - 1$. LandTrendR output consists of a set of breakpoints as well as a fit to the data. The fit is always continuous piecewise linear. The fit is displayed in the results discussed here. Since the fit is piecewise linear, the breakpoints are obvious. The following parameter values are used for LandTrendR in this work: $\upsilon = 0.9$, $\mu = 6$, $\nu = 3$, $\tau = 0.2$, $\varrho = 1.0$.

Success. Fig. 4 presents examples of LandTrendR success. The pixel of Fig. 4(a) is forest, per TimeSync data. In the 12 year period 2000–2012, one significant instance of harvest is recorded in 2010–11. The NDVI trajetory stays stable until 2010, and then a sharp fall is seen in the NDVI values in 2010–11, matching the harvest. LandTrendR captures the trajectory accurately, indicating stability until 2010, a sharp loss in vegetation cover in 2010, followed by gradual recovery.



(a) Forest, Harvest in 2010–11



(b) Agriculture → NVA, Harvest in 2002–03

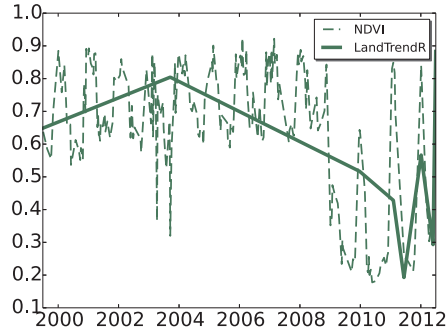
Fig. 4. LandTrendR success.

Fig. 4(b) has a trajectory with mean 0.3–0.4, high variance until 2002. After 2002, the NDVI values are almost constant, close to 0.1, indicating absence of any vegetation cover. TimeSync data indicates that this pixel is an agricultural area that is cleared in 2002–03, and used for nonvegetated anthropogenic purposes starting 2003. The trajectory and the TimeSync information are thus in agreement. LandTrendR indicates some variation in trend pre-2002 and then successfully captures the 2002–03 harvest.

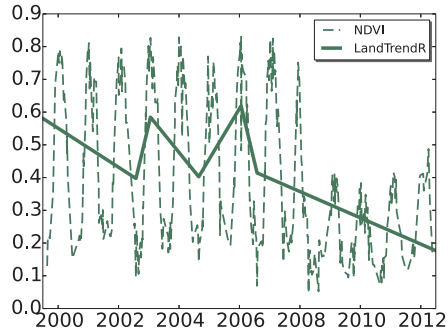
Failure. The two pixels of Fig. 5 present instances of LandTrendR failure. Per TimeSync data, Fig. 5(a) is a forest. The trees are harvested sometime in between 2008–10, and the land is subsequently used for agricultural purposes. The NDVI trajectory exhibits totally different mean and variance before and after 2009. In 2009, a sharp drop in NDVI values is seen, confirming the TimeSync information. The LandTrendR approximation shows an overall decrease in NDVI values between 2004 and 2010, but there is no breakpoint at the 2009 drop. The breakpoints that it does predict appear to be mostly false alarms.

Fig. 5(b) is for a nonforest vegetation area. TimeSync states a mechanical change in 2008–10, after which the area is used for nonvegetated anthropogenic purposes, consistent with the NDVI trajectory: the mean NDVI value has a sharp drop in 2008, and the trajectory after that also has much lower variance. LandTrendR senses a decrease in vegetation cover but there is no breakpoint at the time where the sharp drop occurs. The trend, in general, does not match the trajectory.

Sensitivity. LandTrendR extensively harnesses least squares fitting. The determination of the initial breakpoint set is based on iteratively finding indices where deviation of the least squares fit to the data in certain intervals is maximum (cf. Step 2). Since least squares fit can be easily affected by outliers, this methodology for initial breakpoint set construction is also prone to being affected by outliers. In particular, sometimes points on only one side of a disturbance make it into $\overline{S}$, the sequence of initial vertices, and even these may get dropped in reaching



(a) Forest → agriculture, Mechanical in 2008–10

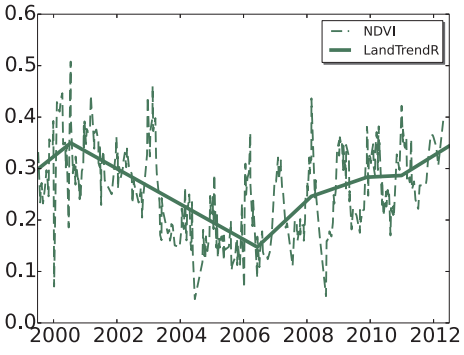


(b) NFV → NVA, Mechanical in 2008–10

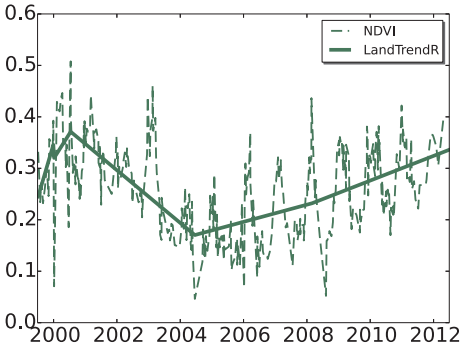
Fig. 5. LandTrendR failure.

the sequence V^*. This makes LandTrendR's success in breakpoint determination sensitive to outliers.

Fig. 6 shows the final outcomes of LandTrendR for two values of the parameter υ. TimeSync states nonforest vegetation type for this pixel prior to 2003, a mechanical event in 2003–04, conversion of land cover type to urban after that, a second mechanical event in 2004–05, and continued urbanized land cover for the rest of the time. The NDVI values start quite small (0.3) and stay low throughout. The changes are not evident in the trajectory, although a gradual decline in NDVI values is seen from 2002 to 2006, and less noise is seen after 2006. Fig. 6(a) corresponds to $\upsilon = 0.8$. LandTrendR gets the overall approximation to the trend correct but misses both the breakpoints for $\upsilon = 0.8$. Fig. 6(b) corresponds to $\upsilon = 0.9$. For $\upsilon = 0.9$, a breakpoint is placed in 2004–05 though the 2003–04 breakpoint is still missing.



(a) $\upsilon = 0.8$



(b) $\upsilon = 0.9$

Fig. 6. LandTrendR sensitivity.

Of relevance to the polyalgorithm are also the parameters μ and ν. With smaller values of these parameters, 'localization' of breakpoints appears to be less prevalent.

6. Breaks for additive and seasonal trend (BFAST)

BFAST (Verbesselt et al., 2010a) decomposes the given time series iteratively into three components: trend, seasonal, and noise. BFAST computes and evaluates least squares fits in windows of increasing size. Qualitatively, (i) first the possibility of there being any structural change in the given time series is determined by computing the partial sums of residuals of least squares fits in windows (OLS-MOSUM). The limiting process of these partial sums is the increments of a Brownian bridge process (Chu et al., 1995). If the observations do have a structural change, an ordinary linear least squares fit will result in large residuals and, hence, in large partial sums. Therefore, the occurrence of large values in the process is an indication of the presence of a structural change — this probability being calculated from the Brownian bridge table. (ii) If a structural change is indicated, a search for change location is done. *Each* interior time point t is considered a breakpoint (change location) candidate. A recursive residual is the error at time t_j from the linear least squares fit over the window $[t_i, ..., t_{j-1}]$. The breakpoints (change locations) are chosen so as to minimize the sum of squared recursive residuals over all windows in between (omitting) the breakpoints. This is done for both trend and seasonal components of the time series, consecutively.

Algorithm BFAST.

for row r 1 **step** 1 **until** R **do**
for column c 1 **step** 1 **until** C **do**
Let $T = (t_1, ..., t_S)$ be the sequence of given time points and the S-vector u denote the time series data in the column $(D_{rc})_{\cdot b}$, i.e., $u = (D_{rc})_{\cdot b}$. Assume that the general model is of the form

$$u = \mathscr{V} + \mathscr{W} + \epsilon,$$

where $\mathscr{V}$ and $\mathscr{W}$ denote the iteratively computed trend and seasonal components, respectively, present in the data and ϵ is the noise. The trend $\mathscr{V}$ may be piecewise linear and the seasonal component $\mathscr{W}$ may be piecewise harmonic. Let N be the maximum number of iterations, n be the iteration number, and $\mathscr{V}^n$ and $\mathscr{W}^n$ be the trend and seasonal components, respectively, computed at the nth iteration. Let $h \in (0, 1)$ denote the proportion of data points by which two consecutive breakpoints t_i and t_j (including t_1 and t_S) must be separated. Thus $\lceil Sh \rceil \leqslant j-i-1$. Take the length of moving windows to be $\lceil Sh \rceil$, initialize the iteration number $n := 1$, and initialize the seasonal component as $\mathscr{W}^0(T) = (w_1^0, ..., w_S^0)$.

Step 1.1: Determine the possibility of breakpoints in trend. Eliminate the seasonal component from the data

$$u^n = u - \mathscr{W}^{n-1}(T).$$

The ordinary least squares (OLS) estimator for the trend is given as

$\alpha = (X^tX)^{-1}X^tu^n$

where X is the Gram matrix for linear regression given by

$$X = \begin{pmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_S \end{pmatrix}.$$

The prediction error (or residual vector or the OLS residual) is defined as

$E^o = u^n - X\alpha,$

where the superscript 'o' is used to signify the fact that these residuals are OLS regression based. Consider the process defined by the moving sums (MOSUM) of these OLS residuals

$$Q^o = \left\{ \frac{1}{\sigma\sqrt{\lceil Sh \rceil}} \sum_{i=k-\lceil Sh \rceil+1}^{k} E_i^o \right\}_{k=\lceil Sh \rceil}^{S},$$

where σ is the sample standard deviation of all the OLS residuals.

Compute the OLS-MOSUM test statistic

$$\hat{f}^o = \max_{1 \leqslant k \leqslant S-\lceil Sh \rceil+1} |Q_k^o|$$

as the maximum absolute value of this process, then compute the asymptotic critical value of the OLS-MOSUM test using the two-sided boundary-crossing probability

$p_T = P[f^o > \hat{f}^o],$

where p_T is read from the Brownian Bridge table.

A p-value less than a user defined parameter $\tau_{\mathscr{S}} \in (0, 1)$ indicates the presence of breakpoints.

Remark 1. As discussed in (Chu et al., 1995), under the null hypothesis, the OLS-MOSUM process converges in distribution to the increments of a Brownian Bridge process.

Step 1.2: Locate trend breakpoints.

Suppose $p_T \leqslant \tau_{\mathscr{S}}$. To locate the breakpoints, consider all possible partitions of the domain, compute OLS fits for each partition, and settle with a partition that yields minimum squared error.

Let $X_{[i,j]}$ denote the matrix formed from rows i through j of the matrix X, and $\alpha_{[i,j]}$ denote the least squares coefficients computed using the matrix $X_{[i,j]}$ with time points $t_i, \ldots, t_j$, and data $u^n_{[i,j]} = u^n_{[i,\ldots,j]}$. For $i = 1, \ldots, S-\lceil Sh \rceil + 1$, consider each window $[t_i, \ldots, t_{j-1}]$, $i + 2 \leqslant j \leqslant S$, and the linear fit in this window. The recursive residual at point t_j is then defined as the weighted prediction error

$$E_{ij}^r = \frac{u^n_{[j,j]} - X_{[j,j]}\alpha_{[i,j-1]}}{\sqrt{1 + X_{[j,j]}(X^t_{[i,j-1]}X_{[i,j-1]})^{-1}X^t_{[j,j]}}}.$$

The superscript 'r' is used to signify the fact that the process/statistic is recursive residual based.

Suppose a breakpoint has been found at t_i. Then the cost of placing the next breakpoint at t_k is calculated as the accumulated sum of squared recursive residuals in the interval $[t_i, t_{k-1}]$, i.e.,

$$\rho_{ik} = \sum_{j=i+2}^{k-1} (E_{ij}^r)^2.$$

All possible positions for the breakpoints can thus be calculated by considering the moving sums of squared recursive residuals, i.e., the process defined by

$$Q^r = \left\{ \left\{ \sum_{j=i+2}^{k} (E_{ij}^r)^2 \right\}_{k=i+2}^{S} \right\}_{i=1}^{S-\lceil Sh \rceil+1}.$$

Given the number μ of desired interior breakpoints, let $k_1, \ldots, k_\mu$ be integers such that $k_{i+1} - k_i > \lceil Sh \rceil$, $k_1 > \lceil Sh \rceil + 1$, and $k_\mu < S - \lceil Sh \rceil$. Determine $K = (1, k_1, \ldots, k_\mu, S)$ to minimize the moving sums of squared recursive residuals

$$\sum_{i=3}^{k_1-1} (E_{1,i}^r)^2 + \sum_{i=k_1+2}^{k_2-1} (E_{k_1,i}^r)^2 + \sum_{i=k_2+2}^{k_3-1} (E_{k_2,i}^r)^2 + \cdots + \sum_{i=k_\mu+2}^{S} (E_{k_\mu,i}^r)^2.$$

Then $(t_{k_1}, \ldots, t_{k_\mu})$ are the interior breakpoints in the trend component.

Remark 2. The breakpoints $t_1, t_{k_1}, \ldots, t_{k_\mu}, t_S$ are optimal in the sense of the above moving sums of squared recursive residuals criterion.

Remark 3. If $p_T > \tau_{\mathscr{S}}$, then there are only two breakpoints (t_1 and t_S) and no interior breakpoints. So this step is skipped and there is simply one linear fit over the entire domain $[t_1, t_S]$ (Step 1.3).

Step 1.3: Let $k_0 = 1$, $k_{\mu+1} = S$, and $I_0 = [t_{k_0}, t_{k_1})$, $I_1 = [t_{k_1}, t_{k_2})$, $\ldots, I_\mu = [t_{k_\mu}, t_{k_{\mu+1}}]$. For each interval I_i, determine the linear regression coefficients

$\gamma^i = (X^t_{[k_i,k_{i+1}]}X_{[k_i,k_{i+1}]})^{-1}X_{[k_i,k_{i+1}]}u^n_{[k_i,k_{i+1}]}$

and construct the (discontinuous) piecewise linear fit

$\mathscr{S}^n(t) = \sum_{i=0}^{\mu} \Gamma^i(t),$

where

$$\Gamma^i(t) = \begin{cases} \gamma_0^i + \gamma_1^i t, & t \in I_i, \\ 0, & \text{otherwise.} \end{cases}$$

Let $\mathscr{S}^n(T) = (v_1^n, \ldots, v_S^n)$ be the sequence of values estimated at $t_1, \ldots, t_S$ using this piecewise linear fit.

Step 2.1: Determine the possibility of breakpoints in seasons.

Eliminate the estimated trend component from the observed data

$\tilde{u}^n = u - \mathscr{S}^n(T).$

The Gram matrix for the seasonal (harmonic) component is given by

$$Y = \begin{pmatrix} 1 & \sin t_1 & \cos t_1 & \cdots & \sin Kt_1 & \cos Kt_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \sin t_S & \cos t_S & \cdots & \sin Kt_S & \cos Kt_S \end{pmatrix},$$

where K is the degree of the trigonometric polynomial used for regression. The trigonometric regression coefficients for the seasonal component are computed as

$$\beta = (Y^t Y)^{-1} Y^t \tilde{u}^n.$$

The prediction error for this fit is defined as

$$E^o = \tilde{u}^n - Y\beta.$$

The OLS-MOSUM process for these errors is given by

$$Q^o = \left\{ \frac{1}{\sigma\sqrt{\lceil Sh \rceil}} \sum_{i=k-\lceil Sh \rceil+1}^{k} E_i^o \right\}_{k=\lceil Sh \rceil}^{S},$$

and the OLS-MOSUM test statistic is

$$\hat{g}^o = \max_{1 \leqslant j \leqslant S-\lceil Sh \rceil+1} |Q_j^o|.$$

The two-sided boundary-crossing probability

$$p_S = P[g^o > \hat{g}^o]$$

is read from the Brownian Bridge table.

A p-value less than a user defined parameter $\tau_{\mathscr{W}} \in (0, 1)$ indicates the presence of seasonal breakpoints.

Step 2.2: Locate seasonal breakpoints.

Suppose $p_S \leqslant \tau_{\mathscr{W}}$. Using the same notation as for the trend breakpoints,

$$E_{ij}^r = \frac{\tilde{u}_{[j,j]}^n - Y_{[j,j]}\beta_{[i,j-1]}}{\sqrt{1 + Y_{[j,j]}(Y_{[i,j-1]}^t Y_{[i,j-1]})^{-1} Y_{[j,j]}^t}}$$

is the recursive residual at time t_j, obtained by trigonometric regression in the time window $[t_i, t_{j-1}]$.

Given the number ν of desired seasonal interior breakpoints and a minimum number of data points separating breakpoints (as for the trend), let $l_1, \ldots, l_\nu$ be integers such that $l_{i+1} - l_i > \lceil Sh \rceil$, $l_1 > \lceil Sh \rceil + 1$, and $l_\nu < S - \lceil Sh \rceil$. Determine $L = (1, l_1, \ldots, l_\nu, S)$ to minimize the moving sums of squared recursive residuals

$$\sum_{i=3}^{l_1-1} (E_{1,i}^r)^2 + \sum_{i=l_1+2}^{l_2-1} (E_{l_1,i}^r)^2 + \sum_{i=l_2+2}^{l_3-1} (E_{l_2,i}^r)^2 + \cdots + \sum_{i=l_\nu+2}^{S} (E_{l_\nu,i}^r)^2.$$

Then $(t_{l_1}, \ldots, t_{l_\nu})$ are the interior breakpoints in the seasonal component.

Remark 4. If $p_S > \tau_{\mathscr{W}}$, then there are only two breakpoints (t_1 and t_S) and no interior breakpoints. So this step is skipped and there is simply one trigonometric polynomial fit over the entire domain $[t_1, t_S]$ (Step 2.3).

Step 2.3: Let $l_0 = 1$, $l_{\nu+1} = S$, and $J_0 = [t_{l_0}, t_{l_1})$, $J_1 = [t_{l_1}, t_{l_2})$, $\ldots, J_\nu = [t_{l_\nu}, t_{l_{\nu+1}}]$. For each interval J_j determine the trigonometric polynomial regression coefficients

$$\delta^j = (Y_{[l_j, l_{j+1}]}^t Y_{[l_j, l_{j+1}]})^{-1} Y_{[l_j, l_{j+1}]} \tilde{u}_{[l_j, l_{j+1}]}^n$$

and construct the (discontinuous) piecewise trigonometric polynomial

$$\mathscr{W}^n(t) = \sum_{j=0}^{\nu} \Delta^j(t), \text{ where} \Delta^j(t) =$$

$$\begin{cases} \delta_1^j + \sum_{k=1}^{K} \delta_{2k}^j \sin kt + \delta_{2k+1}^j \cos kt, & t \in J_j, \\ 0, & \text{otherwise.} \end{cases}$$

Let $\mathscr{W}^n(T) = (w_1^n, \ldots, w_S^n)$ be the sequence of values estimated at $t_1, \ldots, t_S$ using this piecewise trigonometric polynomial approximation.

Step 3: Compare the breakpoints between iterations $n-1$ and n.

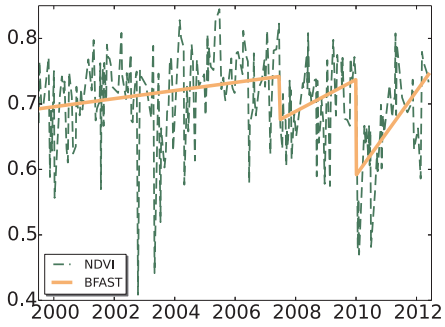
If the Hamming distance between the two breakpoint vectors $(t_{k_1}, \ldots, t_{k_\mu}, t_{l_1}, \ldots, t_{l_\nu})$ at iterations $n-1$ and n is less than some defined tolerance or the number of iterations has reached N, then exit. Otherwise, increment the iteration number n and repeat Steps 1.1 to 3.

end

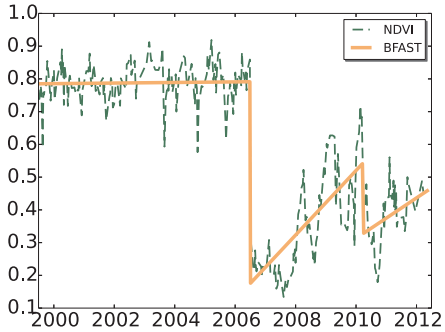
end

The complexity of BFAST is $\mathcal{O}(S^3)$, with the calculation of all the recursive residuals being $\mathcal{O}(S^3)$ and dynamic programming to find the optimal breakpoint sequence being $\mathcal{O}(S^2)$. While the experiments so far assure that the method captures the linear trend correctly, its ability to capture phenological (seasonal) changes has not been studied sufficiently yet. Like LandTrendR, BFAST offers a fit to the data as well as a set of breakpoints. The number of breakpoints is fixed, though, and the piecewise linear fit offered is possibly discontinuous (an advantage). The following parameters are used for BFAST in this work: $K = 1$, $\mu = \nu = 2$, $h = 0.15$, $\tau_{\mathscr{V}} = \tau_{\mathscr{W}} = 0.05$, $N = 2$.

Success. Fig. 7 presents two examples of BFAST's success, the first one being a forest pixel and the second one a nonforest pixel. Fig. 7(a) has two instances of fire — one in 2006–07, and another in 2009–10, both seen in the NDVI trajectory as well. The occurrence of these events is well sensed by BFAST.

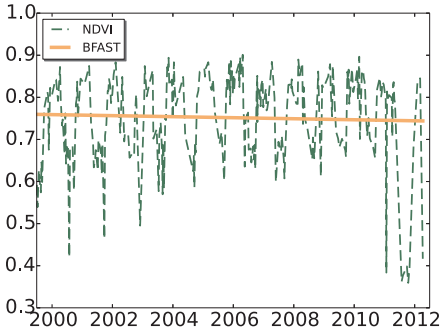


(a) Forest, Fire in 2006 and 2009.

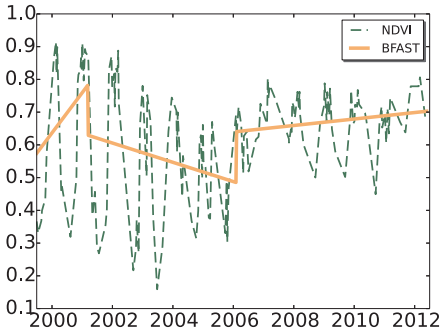


(b) Forest → NVA, Mechanical in 2006 and 2011.

Fig. 7. BFAST success.

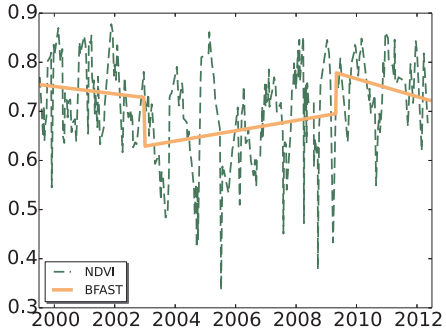


(a) Forest, Harvests in 2002 and 2005.

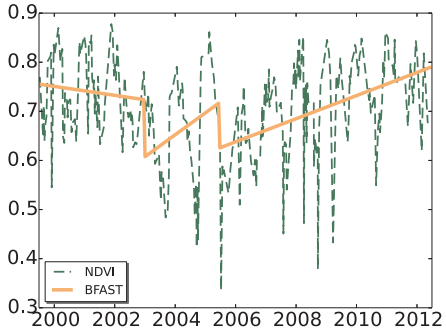


(b) Forest → NFV, Mechanical in 2003–04.

Fig. 8. BFAST failure.



$K = 1$.



$K = 2$.

Fig. 9. BFAST sensitivity.

Fig. 7(right) depicts the functioning of BFAST on forest as well as nonforest land covers. Specifically, this pixel is a forest initially, a mechanical activity causes a sudden loss in vegetation cover in 2006–07, and there is nonvegetated anthropogenic (NVA) usage following that. Based on the NDVI trajectory, some recovery in leaf cover appears to have taken place post2007. A second mechanical event is known to have happened in 2011–12. BFAST is able to capture both events and also replicates the trends in each of the three segments (separated by these breakpoints) correctly. These results endure after minor changes in the parameters.

Failure. Some examples where BFAST fails are presented here. However, such instance are rather rare. In general, it is difficult to find instances where BFAST fails to capture events and/or trends in forests, its main limitation being the *a priori* chosen number of breakpoints. Specifically, if the number of breaks in the time series is more than the number of breaks specified by the user, only the strongest breaks are captured.

Fig. 8(a) corresponds to a forest pixel. There is a harvest in the year 2011–12, reflected in the NDVI trajectory as well. BFAST is not able to capture this change. If two harmonics are used (instead of one) or if the breakpoint spacing proportion h is reduced to $h = 0.10$, this change is captured.

Fig. 8(right) represents a pixel that undergoes a mechanical change in 2003–04, going from forest until 2003 (prior to change) to nonforest vegetation after the change. The NDVI trajectory reflects changes in trend and periodicity; two breakpoints are evident — one in 2004 and another in 2006. BFAST places breakpoints in 2001 and 2006, neither agreeing with TimeSync data. However, the 2006 breakpoint does agree with the NDVI trajectory. This outcome did not change with slight variations in parameters.

Sensitivity. BFAST is mostly able to capture the trends well. For the pixels where it fails, the output for some (not all) of them appears to improve by using more harmonics (K, cf. Step 2.1). Fig. 9 corresponds to a forest pixel known to have been harvested twice — in 2002 and 2005. When published parameters are used, BFAST detects the first harvest but misses the second one. Using two harmonics, the occurrence as well as the timing of both harvests are sensed accurately. Another instance of failure/parameter sensitivity was discussed in Fig. 8(a): for this pixel the breakpoint location is too close to the right boundary of the domain. In such situations, the default value of h prevents the (correct) detection of breakpoints.

7. Prospects for a viable polyalgorithm

Once the algorithms have been implemented, the next step is to develop a procedure to choose the most appropriate outcome for the input. One possibility is to draw a consensus between algorithm outcomes: if more than half the algorithms being used predict breakpoint sets that are in proximity of each other, one of those algorithms gets chosen as the most appropriate algorithm. To this end, a method to measure the distance between the breakpoint sets is needed. Hausdorff distance from topology is used here.

Consider the set of real numbers R and the usual distance function $\overline{d}(x, y) = |x-y|$ defined on R. Suppose $A = \{a_1, ..., a_l\}$ is the set of breakpoints from algorithm $\mathscr{A}$, and $B = \{b_1, ..., b_m\}$ is the set of breakpoints from algorithm $\mathscr{B}$. Then A and B are subsets of R. Using the distance function $\overline{d}(x, y)$, define the distance from a point $a \in A$ to the set B as

$$\hat{d}(a, B) = \inf\{\overline{d}(a, b) | b \in B\}.$$

Then define the distance from set A to set B as

$$d(A, B) = \sup\{\hat{d}(a, B) | a \in A\}.$$

Note that d is not a metric since possibly $d(A, B) \neq d(B, A)$. Hausdorff distance is defined as

$$H(A, B) = \max\{d(A, B), d(B, A)\},$$

and H is a metric.

If every point in A happens to be close to some point B and vice versa, the Hausdorff distance is small. If some point in A is very far from every point in B or vice versa, the Hausdorff distance is large.

Some other facts important in designing the procedure are: (i)

EWMACD works based on training data. If the training data is not from a period of stability, EWMACD outcomes cannot be trusted. In general, a prior knowledge of change during the training period is not available. (ii) Since Hausdorff distance is symmetric, H only determines the *pair* of algorithms that are closest to each other; some method to zero in on a single final algorithm is needed. (iii) The case that an algorithm predicts stability (correctly or incorrectly) for a pixel will often arise, so the empty breakpoint set special case must be handled.

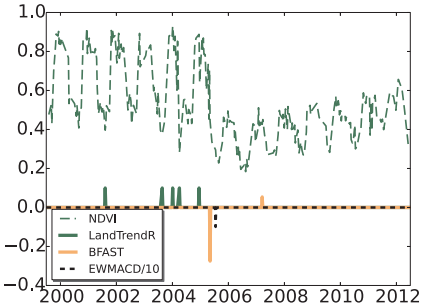
In this early work, the following overall procedure for an LULCC polyalgorithm is used:

(1) Run the three algorithms on the input.
(2) For EWMACD, consider only isolated breakpoints. Specifically, during periods of stability, $f_i = f_{i-1}$. For any i, if $f_i \neq f_{i-1}$, check the previous, say 50, time points. If $f_{i-1} = f_j, j = i-50, \ldots, i-2$, then use t_i as a breakpoint. Otherwise, t_i is only a part of an ongoing recovery or loss period, does not mark a new event, and is ignored.
(3) If BFAST predicts a change during the training period of EWMACD, EWMACD results are not used for that pixel (only BFAST and LandTrendR are used).
(4) If $A = \emptyset$ and $B \neq \emptyset$, $d(A, B)$ is undefined and not used. If $A \neq \emptyset$ and $B = \emptyset$, $d(A, B) = \infty$. If $A = B = \emptyset$, $d(A, B) = 0$. The interpretation of this is that the two algorithms agree totally on breakpoints for this pixel.
(5) If $d(A, B) \ll d(B, A)$, by the principle of parsimony, choose $\mathscr{A}$ over $\mathscr{B}$.
(6) If $d(A, B) > d_\tau, \forall A, B$, where d_τ is a user defined threshold, then no consensus has been found between the algorithms, and the pixel is declared to be stable. Due to the limited number of algorithms included at present, and lack of control over the number of breakpoints for BFAST and LandTrendR, $d_\tau = 13$ (the number of years in the data) is used. This is because two algorithms that agree and have found a breakpoint correctly, may have temporally distant false alarms, resulting in either of them not getting selected. As the polyalgorithm evolves, lower values of d_τ can be used.

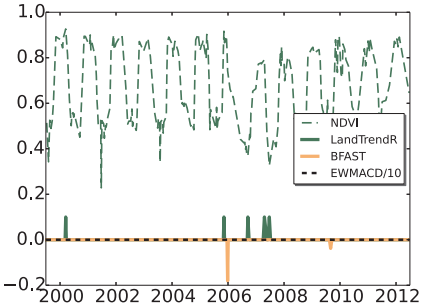
7.1. Results

Each component algorithm produces a different kind of outcome (flags for disturbance, continuous fit, discontinuous fit). In the rest of this work, only the breakpoints produced by each algorithm are considered (the focus of this work being correct breakpoint placement); any 'fit' or 'recovery period information' (originally included in the figures in Sections 3–5) is not included here. For BFAST $K = 2$ is used. For the formulae appearing in the descriptions in the rest of this section, the breakpoint set produced by EWMACD is denoted by E, that by BFAST by B, and that by LandTrendR by L. For BFAST, the height of a jump corresponds to the signed magnitude of the corresponding discontinuity in the proposed model. For EWMACD, the height of the jump corresponds to the signed magnitude of the flag predicted by EWMACD at the corresponding location. For LandTrendR, the model is continuous. So a value of $+1$ is used at its breakpoints for display purposes.

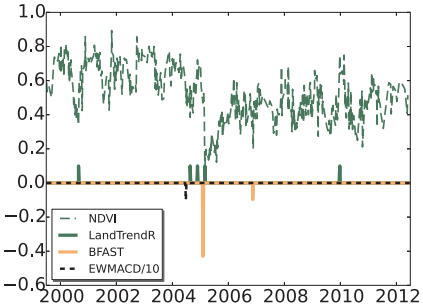
Fig. 10 displays instances of polyalgorithm success. The pixel of Fig. 10(a) has a stable NDVI trajectory until 2005, suffers a sudden drop in NDVI to almost no vegetation in 2005, and has small but gradually increasing NDVI values thereafter. Based on this trajectory, there should be exactly one breakpoint in 2005. Per TimeSync data, this pixel is a forest that is cleared of vegetation in 2005–06 and is used for anthropogenic purposes after that, agreeing with the TimeSync data and NDVI trajectory. On running the polyalgorithm for this pixel, EWMACD produces one breakpoint, precisely at the location where the drop in NDVI occurs. BFAST produces two breakpoints (as requested), one at the NDVI drop location and another in 2007. LandTrendR produces four breakpoints, the first three being false alarms between 2001 and 2005, and the fourth one being close to the drop location. The breakpoint-sets



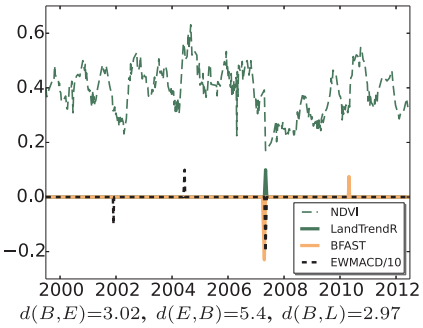
(a) Forest → NVA, Mechanical in 2005.



(b) Forest, Harvest in 2007.



(c) NFV → NVN, Flood in 2004.



(d) Nonforest vegetated, Fire in 2007.

Fig. 10. Polyalgorithm outcomes.

are: $E = \{2006.03\}$, $B = \{2005.85, 2007.72\}$, $L = \{2002.1, 2004.2, 2004.5, 2004.8, 2005.5\}$. The $d(\cdot,\cdot)$ values are displayed along with the figures. Since $d(E, B) = 0.18$ is the smallest one, EWMACD gets chosen as the final algorithm and the final breakpoint is January 10th, 2006, quite in

agreement with both the trajectory as well as TimeSync data. Note that both BFAST and LandTrendR also catch the break in 2005–06.

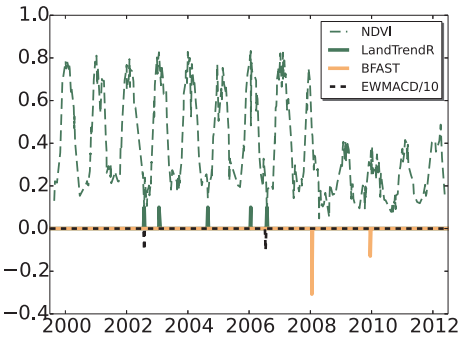
Fig. 10(b) has a stable trajectory until 2006, a sudden drop in NDVI in 2006, with gradual recovery in the subsequent years. There is one breakpoint in the trajectory. Per TimeSync data, this pixel is a forest, with a single harvest in 2007–08. EWMACD assesses this pixel as stable. LandTrendR has a breakpoint in 2000, and multiple breakpoints in 2006 to 2008. BFAST has breakpoints in 2006 and 2010. The breakpoint sets are: $E = \{\}$, $B = \{2006.5, 2010.17\}$, and $L = \{2000.7, 2006.36, 2007.2, 2007.8, 2007.9\}$. The $d(\cdot,\cdot)$ values are displayed along with the figure. BFAST gets selected as the final algorithm, labeling 2006 and 2010 (a false alarm) as breakpoint locations.

Fig. 10(c) has a drop in NDVI in 2005, from vegetated to almost zero vegetation. The NDVI recovers in 2005–07. The new vegetation NDVI has a smaller mean. For this trajectory, two breakpoints are expected: one in 2005, and another in 2007. Per TimeSync data, this pixel is initially covered with nonforest vegetation (NFV), experiences a flood in 2004, and has nonvegetated natural (NVN) land cover thereafter. TimeSync does not note the end of recovery period/beginning of the stable period. Each of the three algorithms detects the flooding event, with their breakpoint placement for the flood being not exactly the same but in close proximity to each other. EWMACD places one breakpoint in 2004, and none in 2006. BFAST produces two breakpoints, the first in 2005, and the second in 2007. BFAST output is thus most in agreement with the NDVI trajectory. LandTrendR places multiple breaks in 2004–05, one each in 2000 and 2010, and none in 2006–07. The polyalgorithm selects EWMACD as the final algorithm. EWMACD's selection can be attributed to (i) fewest number of breakpoints, and (ii) each of its breakpoints being close to some breakpoint of LandTrendR. The outcome of each of the three component algorithms as well as the polyalgorithm agrees with TimeSync data, but BFAST offers the best match to the trajectory. This pixel with flooding does not reflect the general situation with flooding, where all the algorithms perform poorly.

The low NDVI values in Fig. 10(d) indicate relatively low vegetation cover. There is one sharp drop in 2007, recovery until 2010, and stability thereafter. TimeSync data classifies this pixel as one covered with nonforest vegetation, with an instance of fire in 2007. The trajectory and TimeSync are in agreement, except that TimeSync does not record the end of a recovery period. EWMACD detects four breakpoints, BFAST two, and LandTrendR two. The three algorithms agree only on 2007 being the breakpoint, which is the desired outcome. The polyalgorithm chooses LandTrendR as the final algorithm.

Sometimes, a 'majority' of the component algorithms (two or more out of three here) capture the same set of incorrect breakpoints leading to polyalgorithm failure. Per TimeSync, the pixel of Fig. 11(a) is initially covered with nonforest vegetation, cleared in 2008–10, and used for anthropogenic purposes after that. Per the NDVI trajectory, there is a major lasting change in 2008. BFAST detects the occurrence of this change correctly. However, both EWMACD and LandTrendR detect changes in 2002–03 and 2006–07 (in addition to more changes signalled by LandTrendR). Due to this, the EWMACD outcome gets finally chosen by the polyalgorithm. Clearly, the breakpoints based on this final selection (EWMACD) are not correct.

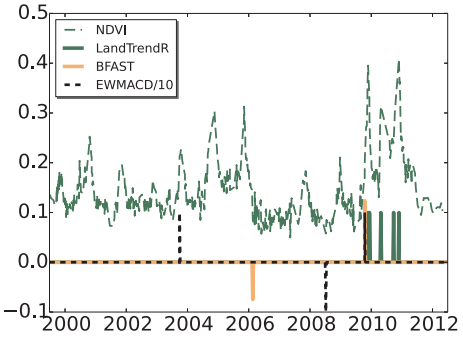
Fig. 11(b) shows how an algorithm's localized output can lead to the selection of a suboptimal algorithm. The NDVI trajectory for this pixel has four breakpoints: 2004 (beginning of a recovery period), 2006 (an abrupt drop in NDVI), 2010 (another recovery period), and 2011 (another abrupt drop in NDVI), TimeSync records specify a fire in 2006, and no other event. Ideally, a change detection algorithm would have four breakpoints. EWMACD places breakpoints in 2004, 2008, and 2010. It misses the events in 2006 and 2011, because these events occur amidst unstable periods. In all, EWMACD disagrees with TimeSync information, places breakpoints at two out of four locations expected based on NDVI trajectory. BFAST, with the allowed two breakpoints, places breaks in 2006 and 2010. LandTrendR, on the other hand, with



$d(B,E)=3.4$, $d(E,B)=5.5$, $d(B,L)=3.4$

$d(L,B)=5.47$, $d(L,E)=1.9$, $d(E,L)=0.07$

(a) NFV → NVA, Mechanical in 2008.



$d(B,E)=2.36$, $d(E,B)=2.39$, $d(B,L)=3.76$,

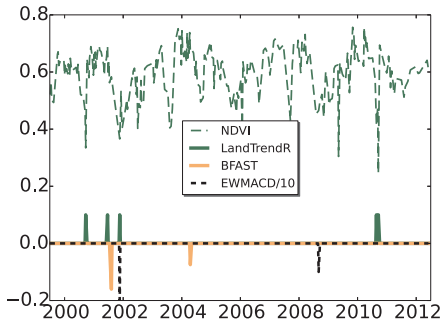
$d(L,B)=1.115$, $d(L,E)=1.118$, $d(E,L)=6.15$

(b) NFV, Fire in 2006.

Fig. 11. Current limitations of the polyalgorithm.

its binary search pattern of stencil choosing, places multiple breakpoints in a very small period 2009–11. Overall, all three algorithms agree on the breakpoint in 2010, but only BFAST catches the fire in 2006–07, only EWMACD catches the beginning a of recovery period in 2004, and only LandTrendR catches the event in 2011. With all breakpoints of LandTrendR accumulated in one region, and one breakpoint (2010) each of EWMACD and BFAST close to all these breakpoints, $d(L, B)$ and $d(L, E)$ are small numbers and the polyalgorithm selects LandTrendR for the final outcome, disagreeing with TimeSync information, and agreeing with the trajectory only on the 2010–12 part. Note that none of the three algorithms captures all four breakpoints suggested by the trajectory (although BFAST got its limit of two correct).

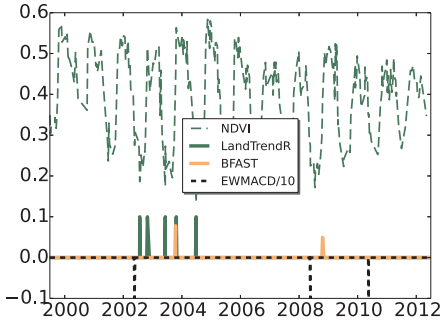
Finally, instances where the TimeSync data differs from the NDVI trajectory are presented. In Fig. 12(a), all three algorithms yield outcomes that disagree with TimeSync information. The NDVI trajectory appears mostly stable except two instances of NDVI drop — in 2002 and 2008. TimeSync does not record any event during the time period under consideration. Per TimeSync, this pixel is being used for nonvegetated anthropogenic purposes (although the NDVI values appear to be high for the pixel to not have any vegetation). Each of the component algorithms predicts breakpoints. Due to BFAST's outcome, EWMACD's selection for the final algorithm is ruled out. (This explains the ∞ values of $d(E, \cdot)$ and $d(\cdot,E)$ displayed with the figure. Note that, this implies that even if EWMACD had correctly captured the stability, it would not get selected as the final algorithm because of BFAST's outcome.) BFAST gets selected as the final algorithm. In Fig. 12(right), TimeSync reports mechanical change in 2009–10 but no such change is evident in the



$d(B,E)=\infty$, $d(E,B)=\infty$, $d(B,L)=2.4$

$d(L,B)=6.4$, $d(L,E)=\infty$, $d(E,L)=\infty$

(a) Nonvegetated anthropogenic, stable.



$d(B,E)=1.4$, $d(E,B)=1.6$, $d(B,L)=4.3$

$d(L,B)=1.2$, $d(L,E)=2.1$, $d(E,L)=5.8$

(b) Nonvegetated anthropogenic, Mechanical in 2009.

Fig. 12. NDVI trajectory vs. TimeSync records.

NDVI trajectory. On the other hand, the trajectory exhibits three distinct trends: descending NDVI values from 2000 to 2003, a flat trend from 2003 to 2008, and an ascent in NDVI values from 2008–12. Thus, breakpoints in 2003 and 2008 are appropriate. EWMACD and BFAST place breaks 2002–04 and 2008–09. LandTrendR is chosen as the final algorithm. A third instance of disagreement between TimeSync and NDVI trajectory was discussed in Fig. 11(b).

7.2. Discussion

Estimates of true positives (TPs, instances where an algorithm correctly identifies a breakpoint) and false negatives (FNs, instances where an algorithms misses a breakpoint) using TimeSync data as reference are presented in Table 2. Four specific types of events are considered:

Table 2
Breakpoint detection relative to TimeSync data.

		EWMACD	LTR	BFAST	PolyAlg.
Harvest	TP	271	304	368	340
(558 pixels)	FN	340	307	243	275
	FP	204	913	310	368
Mechanical	TP	33	44	49	46
(58 pixels)	FN	39	28	23	26
	FP	83	260	102	123
Flood	TP	8	11	9	8
(19 pixels)	FN	13	10	12	13
	FP	196	581	226	261
Fire	TP	79	72	88	82
(130 pixels)	FN	54	61	45	51
	FP	25	94	30	39

Table 3
Percentage of algorithm selection.

	EWMACD (%)	LandTrendR (%)	BFAST (%)	None (%)
Harvest	32.8	50.5	14.9	1.7
Mechanical	25.8	25.8	46.5	1.7
Flood	42	26.3	31.6	0
Fire	27.7	20.8	49.2	2.3

harvest, mechanical, flood, and fire. The total number of pixels that contain these events (not the same as the total number of events) is listed in parentheses beside the event name. Table 3 shows the percentage of algorithm selection in the current polyalgorithm.

BFAST and LandTrendR have high false positive (FP) rates. In particular, for BFAST, the number of breakpoints the algorithm must calculate is fixed (ν, μ, user defined, default to two or none in this work), so BFAST is expected to produce one false alarm for each pixel that happens to have only a single event. This explains the overall high false alarm count for the algorithm. For the current set of parameters LandTrendR is also characterized by several false alarms for any given trajectory. EWMACD is flexible in terms of the number of breakpoints, thereby producing fewer FPs. The polyalgorithm naturally inherits false alarms from the component algorithms. Preliminary experiments to allow flexible values of ν, μ for BFAST and μ for LandTrendR have shown encouraging results, but a reliable strategy to adopt this feature is currently not there. On a different note, for short term trajectories, the available TimeSync data contains only major events such as fire, harvest, and the like (and, sometimes, does not agree with the NDVI trajectory (Fig. 12)). Each of the three algorithms frequently registers the beginning of any new trend in trajectory as a breakpoint as well. Since 'nonmajor-changes' such as beginning or end of recovery period are not included in TimeSync data, these breakpoints often get (incorrectly) counted towards false alarms. With such limitations, the presented FP numbers are a rather unfair assessment.

In case of BFAST, when there exists an ongoing gradual event in the beginning of the trajectory (e.g., a harvest from 2000–02), BFAST places a breakpoint at the end of this change, or, in other words, at the beginning of the next (recovery or stability) trend (so, in 2002 or 2003 for the harvest that extended from 2000–02). This breakpoint is correct with respect to the trajectory. However, for this example, TimeSync records 2000–02 as the change years. So while checking against TimeSync data, no change is found in the years on record and BFAST gets a false negative. Consequently, BFAST's true positives are slightly undercounted and false negatives are slightly overcounted. The same is true for the polyalgorithm's false negatives/true positives.

Based on the numbers in the tables, the polyalgorithm demonstrates improved results compared to either EWMACD or LandTrendR alone. However, BFAST turns out to be the highest performing algorithm, even better than the polyalgorithm itself. The results collectively suggest that, for a significant number of instances, (i) EWMACD and LandTrendR miss 'major' events while these events are correctly identified by BFAST, but (ii) EWMACD and LandTrendR still get chosen as the final algorithm by the polyalgorithm. The selection in these instances could potentially be because of (i) both algorithms finding similar sets of incorrect breakpoints and therefore being in consensus, or (ii) situations similar to Figs. 11 and 12(b) — one of the component algorithms yields a clustered set of breakpoints, which also happens to be close to any breakpoint from one of the other component algorithms. Experiments on long term trajectories with the same polyalgorithm strategy have not shown any significant changes in success rates. Including more algorithms will partially alleviate this situation. To robustly combat this situation, the polyalgorithm strategy itself needs to evolve further. Minor variations in polyalgorithm strategy have shown promise but having more algorithms will be beneficial prior to any significant strategy changes.

BFAST vs. polyalgorithm. BFAST is very thorough in its breakpoint search, so its good performance is not surprising. It appears that using BFAST alone may be better than using multiple algorithms and the polyalgorithm. However, BFAST does not scale well (Saxena et al., 2017a,b). Furthermore, the user-defined, fixed number of breakpoints ($\mu = 2$ in this study) that BFAST uses is a significant limitation. For the same stack of images (scene), different pixels may have zero, one, two, or more breakpoints. In addition, for longer lengths of time more breakpoints may be desirable. Two breakpoints are barely enough to capture one loss plus the subsequent recovery. Eventually, therefore, other algorithms that allow on-the-fly determination of breakpoints and are also scalable will need to be included as component algorithms. BFAST can instead be used, for example, in subintervals of the dataset to validate breakpoints produced by other component algorithms, or, to check stability in the training period of algorithms dependent on training data. The polyalgorithmic approach appears to be the most promising way of change detection over a variety of land covers, but a viable polyalgorithm remains to be worked out.

Dependence on parameters. The present results are based on a fixed set of parameters (the published parameters). In some (to many) instances, variation of input parameters for any one algorithm could well cause output variability comparable to that between algorithms. Using larger values of persistance ϖ and L for EWMACD, for instance, make significant differences in the output. Similarly, using $\nu = 0$ for LandTrendR appears to reduce clustering of breakpoints. The current polyalgorithm used the published value $\nu = 0$. The number of breakpoints used for BFAST and LandTrendR, in particular, has significant influence on the polyalgorithm outcomes.

Imbalance in number of breakpoints. Currently, EWMACD has an independent number of breakpoints, BFAST has a user defined, fixed number of breakpoints, while LandTrendR has a user defined upper limit on the number of breakpoints. Different algorithms producing different numbers of breakpoints almost always leads to a significant imbalance in the Hausdorff directional distance based voting. To alleviate this imbalance, one possibility is to run EWMACD, if it produces a minimum number of breakpoints (minimum being decided based on the total number of years in the study), set μ for LandTrendR and $\nu = \mu$ for BFAST equal to it. Preliminary experiments with this approach have shown improved results for BFAST, LandTrendR and the polyalgorithm. However, more algorithms need to be included (at least, to combat the dependence of EWMACD itself on its training period) and outcomes studied further before drawing conclusions.

8. Conclusions and future work

At present, the polyalgorithm offers some improvement over EWMACD or LandTrendR alone, but not over BFAST. Since BFAST is the most rigourous/exhaustive component algorithm, and the polyalgorithm is not as good as BFAST yet, there is surely room for improvement of the polyalgorithm. The experiments carried out in this work are just the beginning of further work towards a viable polyalgorithm. Several directions of research/development must be pursued in this regard.

First, the number of algorithms included here is not sufficient for a robust consensus vote. Two out of three algorithms often miss trends and/or important changes of interest to users. With only three algorithms in the framework, the polyalgorithm also misses these changes/events. At least two more algorithms must be included for robust voting. Algorithms based on splines (e.g., Moisen et al., 2016) and those utilizing spatial information have potential for inclusion.

Second, the current algorithm selection is not exhaustive in data coverage. For example, for trajectories where there is no significant change in trend and the training period of EWMACD is also unstable, only LandTrendR is applicable. An algorithm suitable for such trajectories is needed. Further, most examples presented in this paper are related to vegetated pixels (mainly because NDVI was being used as input). Studying the use of the polyalgorithm to monitor urban land cover changes is one of the next steps and will likely involve utilizing other spectral bands.

Third, in addition to determining the 'best' algorithm for a given pixel, it may also be possible to incorporate parameter tuning/adaptation in the polyalgorithm. For example, for a random sample of pixels with known validated changes, the polyalgorithm may run each algorithm for a range of parameters rather than for the fixed set of (published) parameters. The polyalgorithm will then also suggest good parameter choices for each algorithm on the full data set. Adding such intelligence to any of the algorithms, and to the polyalgorithm, is a significant research challenge.

Fourth, resource allocation is a concern. Parallelism is presently implemented only across pixels (first level) using OpenMP, i.e., multiple cores of a node take different pixels and process them simultaneously. For a fixed pixel, the component algorithms are currently executed serially. With this arrangement, load balancing within individual algorithms is already an issue. If a national level image analysis has to be carried out or if parameter tuning has to be included in the polyalgorithm, parallelism across algorithms (second level) becomes essential. A third layer of parallelism, namely, parallelism within algorithms, is also desirable. Load balancing between these three (coarse to fine grained) levels of parallelism in the polyalgorithm will be challenging.

Finally, the directional Hausdorff distance $d(\cdot,\cdot)$ alone is not sufficient to determine the polyalgorithm outcome. The instances of failure of the current basic strategy are primarily due to an insufficient number of component algorithms, and equally important, due to multiple algorithms producing proximal but incorrect results. To fully circumvent the latter situation, the polyalgorithm strategy needs to evolve further. Possibilities include: (i) including parameter variation (ii) breakpoint acceptance test (for example, residuals for a piecewise linear fit based on proposed breakpoints). (iii) tests for goodness of fit and correlation coefficient, (iv) preemptively checking whether a component algorithm is even suitable for a given time series (e.g., if the training period of EWMACD is not stable, then EWMACD is simply not used for that pixel). Including more algorithms will allow designing more sophisticated strategies.

Given the inconsistent trend predictions between the different algorithms, the sometimes erratic behavior of a given algorithm on a given image stack, the sensitivity to parameters for some algorithms, and the prohibitive execution times for serial codes, there is clearly a need for a parallel polyalgorithm (an intelligent, adaptive union of multiple algorithms). This and earlier work (Saxena et al., 2017a), assessing the scalability and memory footprint, the parameter sensitivity, and the range of applicability of individual algorithms are but first steps toward such a parallel polyalgorithm for hyperspectral Landsat image stacks. In summary, the contributions of this work are (1) a systematic investigation of the issues involved in creating a polyalgorithm; (2) precise mathematical descriptions of BFAST, EWMACD, and LandTrendR; (3) efficient, portable, parallel Fortran 2008 code for all algorithms; (4) directional Hausdorff distance for comparing changes; (5) a unified I/O framework for running and comparing LULCC algorithms.

9. Author contributions

RS wrote the codes and conducted the experiments; RS, LTW, and RHW wrote the paper; EBB, VAT, YZ, and REK assisted in data collection and validation.

Acknowledgements

This work was supported in part by USDA Forest Service Grant 13-JV-11330145-046, NSF Grant CNS-1565314, U.S. Geological Survey Contract G12PC00073, NASA Grants NNX17AI07G and NNX17AI09G, and the Virginia Agricultural Experiment Station and the McIntire-Stennis Program of NIFA, USDA (Project Number 1007054, "Detecting and Forecasting the Consequences of Subtle and Gross Disturbance on

Forest Carbon Cycling"). The authors thank the anonymous reviewers for their insightful feedback on the original manuscript.

References

Agrawal, R., Faloutsos, C., Swami, A., 1993. Efficient similarity search in sequence databases. In: Lomet, D.B. (Eds.), Foundations of Data Organization and Algorithms, pp. 69–84.

Anees, A., Aryal, J., O'Reilly, M.M., Gale, T.J., Wardlaw, T., 2016. A robust multi-kernel change detection framework for detecting leaf beetle defoliation using Landsat 7 ETM + data. ISPRS J. Photogram. Remote Sens. 122, 167–178.

Banner, A., Lynham, T., 1981. Multitemporal analysis of Landsat data for forest cutover mapping — a trial of two procedures. In: Proc. of the 7th Canadian Symposium on Remote Sensing, Winnipeg, Canada, pp. 233–240.

Benedek, C., Shadaydeh, M., Kato, Z., Szirányi, T., Zerubia, J., 2015. Multilayer Markov Random Field models for change detection in optical remote sensing images. ISPRS J. Photogram. Remote Sens. 107, 22–37.

Bouziani, M., Goia, K., He, D.C., 2010. Automatic change detection of buildings in urban environment from very high spatial resolution images using existing geodatabase and prior knowledge. ISPRS J. Photogram. 65 (1), 143–153.

Box, G.E.P., Jenkins, G.M., 1970. Time Series Analysis: Forecasting and Control. San Francisco: Holden Day. (Revised edition published 1976).

Brooks, E.B., Wynne, R.H., Thomas, V.A., Blinn, C.E., Coulston, J.W., 2014. On-the-fly massively multitemporal change detection using statistical quality control charts and Landsat data. IEEE Trans. Geosci. Remote Sens. 52 (6), 3316–3332.

Brooks, E.B., Zhiqiang, Y., Thomas, V.A., Wynne, R.H., 2017. Edyn: dynamic signaling of changes to forests using exponentially weighted moving average charts. Forests 8 (304), 18.

Cai, S., Desheng, L., 2015. Detecting change dates from dense satellite time series using a sub-annual change detection algorithm. Remote Sens. 7, 8705–8727.

Campbell, J.B., Wynne, R.H., 2011. Introduction to Remote Sensing, fifth ed. Guilford Publications.

Chan, K.P., Fu, A.W-C., 1999. Efficient time series matching by wavelets. In: Proc. of the 15th IEEE Int. Conference on Data Engineering (ICDE) 8 pages.

Chen, G., Hay, G.J., Carvalho, L.M., Wulder, M.A., 2012a. Object-based change detection. Int. J. Remote Sens. 33 (14), 4434–4457.

Chen, X., Chen, J., Shi, Y., Yamaguchi, Y., 2012b. An automated approach for updating land cover maps based on integrated change detection and classification methods. ISPRS J. Photogram. Remote Sens. 71, 86–95.

Chu, C-S.J., Hornik, K., Kuan, C.-M., 1995. Mosum tests for parameter constancy. Biometrika 82, 603–617.

Cohen, W.B., Fiorella, M., 1998. In: Elvidge, C.D., Lunetta, R.S. (Eds.), Remote Sensing Change Detection: Environmental Monitoring Applications and Methods. Ann Arbor Press, pp. 89–102.

Cohen, W.B., Fiorella, M., Gray, J., Helmer, E., Anderson, K., 1998. An efficient and accurate method for mapping forest clear cuts in the Pacific Northwest using Landsat imagery. Photogram. Eng. Remote Sens. 64, 293–300.

Cohen, W.B., Yang, Z., Kennedy, R.E., 2010. Detecting trends in forest disturbance and recovery using yearly Landsat time series: 2. TimeSync-Tools for calibration and validation. Remote Sens. Environ. 114 (12), 2911–2924.

Cohen, W.B., Healey, S.P., Zhiqiang, Y., Stehman, S.V., Brewer, C.K., Brooks, E.B., Gorelick, N., Huang, C., Hughes, M.J., Kennedy, R.E., Loveland, T.R., Moisen, G.G., Schroeder, T.A., Vogelmann, J.E., Woodcock, C.E., Yang, L., Zhu, Z., 2017. How similar are forest disturbance maps derived from different landsat time series algorithms. Forests 8 (98). https://doi.org/10.3390/f8040098.

Coppin, P.R., Bauer, M.E., 1994. Processing of multitemporal Landsat TM imagery to optimise extraction of forest cover change features. IEEE Trans. Geosci. Remote Sens. 32, 918–927.

Coppin, P., Jonckheere, I., Nackaerts, K., Muys, B., Lambin, E., 2004. Digital change detection methods in ecosystems monitoring: a review. Int. J. Remote Sens. 25 (5), 1565–1596.

Dietterich, T.G., Kittler, J., Roli, F., 2001. Ensemble methods in machine learning. In: Multiple Classifier Systems. LNCS, vol. 1857. Springer, pp. 1–15.

Douglas, D.H., Peucker, T.K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its carricature. Canad. Cartograph. 10 (2), 112–122.

Duda, R.O., Hart, P.E., 1973. Pattern Classification and Scene Analysis. Wiley, New York.

Fall, S., Niyogi, D., Gluhovsky, A., Pielke Sr., R.A., Kalnay, E., Rochon, G., 2010. Impacts of land use and land cover on temperature trends over the continental United States: assessment using the North American Regional Reanalysis. Int. J. Climatol. 30 (13), 1980–1993.

Fung, T., 1990. An assessment of TM imagery for land cover change detection. IEEE Trans. Geosci. Remote Sens. 28, 681–684.

Fung, T., LeDrew, E., 1987. Application of principal components analysis to change detection. Photogram. Eng. Remote Sens. 53, 1649–1658.

Gil-Yepes, J.L., Ruiz, L.A., Recio, J.A., Balaguer-Beser, A., Hermosilla, T., 2016. Description and validation of a new set of object-based temporal geostatistical features for land-use/land-cover change detection. ISPRS J. Photogram. Remote Sens. 121, 77–91.

Goldstein, T., Osher, S., 2009. The split Bregman method for L1-regularized problems. SIAM J. Imag. Sci. 2 (2), 323–343.

Gomeni, R., Gomeni, C., 1979. AUTOMOD: a polyalgorithm for an integrated analysis of linear pharmacokinetic models. Comput. Biol. Med. 9 (1), 39–48.

Goodwin, N.R., Coops, N.C., Wulder, M.A., Gillanders, S., Schroeder, T.A., Nelson, T., 2008. Estimation of insect infestation dynamics using a temporal sequence of Landsat data. Remote Sens. Environ. 112, 3680–3689.

Häfner, H., Schönauer, W., Weiss, R., 1999. The program package LINSOL — basic concepts and realization. Appl. Numer. Math. 30 (2–3), 213–224.

Hame, T.H., 1986. Satellite image aided change detection. In: Remote sensing-aided forest inventory, Research Notes No. 19, Department of Forest Mensuration and Management, University of Helsinki, Helsinki, Finland, pp. 47–60.

Hansen, M.C., Potapov, P.V., Moore, R., Hancher, M., Turubanova, S.A., Tyukavina, A., Thau, D., Stehman, S.V., Goetz, S.J., Loveland, T.R., Kommareddy, A., Egorov, A., Chini, L., Justice, C.O., Townshend, J.R.G., 2013. High-resolution global maps of 21st-century forest cover change. Science 342, 850–853.

Healey, S.P., Cohen, W.B., Yang, Z., Brewer, C.K., Brooks, E.B., Gorelick, N., Hernandez, A.J., Huang, C., Hughes, M.J., Kennedy, R.E., Loveland, T.R., Moisen, G.G., Schroeder, T.A., Stehman, S.V., Vogelmann, J.E., Woodcock, C.E., Yang, L., Zhu, Z., 2017. Mapping forest change using stacked generalization: an ensemble approach. Remote Sens. Environ. https://doi.org/10.1016/j.rse.2017.09.029.

Huang, C., Goward, S.N., Masek, J.G., Thomas, N., Zhu, Z., Vogelmann, J.E., 2010. An automated approach for reconstructing recent forest disturbance history using dense Landsat time series stacks. Remote Sens. Environ. 114 (1), 183–198.

Hughes, M.J., Kaylor, S.D., Hayes, D.J., 2017. Patch-based forest change detection from landsat time series. Forests 8 (5), 1–22.

Hunter, J., McIntosh, N., 1999. Artificial Intelligence in Medicine. Springer.

Hussain, M., Chen, D., Cheng, A., Wei, H., Stanley, D., 2013. Change detection from remotely sensed images: from pixel-based to object-based approaches. ISPRS J. Photogram. Remote Sens. 80, 91–106.

Iersel, W.V., Straatsma, M., Addink, E., Middlekoop, H., 2018. Monitoring height and greenness of non-woody floodplain vegetation with UAV time series. ISPRS J. of Photogram. Remote Sens. 141, 112–123.

Jensen, J.R., 1983. Urban change detection mapping using Landsat digital data. Am. Cartograph. 81, 127–147.

Jin, S., Yang, L., Danielson, P., Homer, C., Fry, J., Xian, G., 2013. A comprehensive change detection method for updating the National Land Cover Database to circa 2011. Remote Sens. Environ. 132, 159–175.

de Jong, R., Verbesselt, J., Schaepman, M.E., de Bruin, S., 2012. Trend changes in global greening and browning: contribution of short-term trends to longer-term change. Global Change Biol. 18, 642–655.

de Jong, R., Verbesselt, J., Zeileis, A., Schaepman, M., 2013. Shifts in global vegetation activity trends. Remote Sens. 5 (3), 1117–1133.

Joyce, A.T., Burns, G.S., 1981. Evaluation of land cover change detection techniques using Landsat MSS data. In: Proc. of the 7th PECORA Symposium, Sioux Falls, SD, USA (Bethesda, MD: ASPRS), pp. 252–260.

Kennedy, R.E., Cohen, W.B., Schroeder, T.A., 2007. Trajectory-based change detection for automated characterization of forest disturbance dynamics. Remote Sens. Environ. 110, 370–386.

Kennedy, R.E., Yang, Z., Cohen, W.B., 2010. Detecting trends in forest disturbance and recovery using yearly Landsat time series: 1. LandTrendr – temporal segmentation algorithms. Remote Sens. Environ. 114, 2897–2910.

Keogh, E., Chu, S., Hart, D., Pazzani, M., 2001. An online algorithm for segmenting time series. In: Proc. of IEEE Int. Conference Data Mining, pp. 289–296.

Kittler, J., Hatef, M., Duin, R.P.W., Matas, J., 1998. On combining classifiers. IEEE Trans. Pattern Anal. Mach. Intell. 20 (3), 226–239.

Koski, A., Juhola, M., Meriste, M., 1995. Syntactic recognition of ECG signals by attributed finite automata. Pattern Recogn. 28 (12), 1927–1940.

Kriegler, F.J., Malila, W.A., Nalepka, R.F., Richardson, W., 1969. Preprocessing transformations and their effects on multispectral recognition. In: Proc. of the Sixth Int. Symposium on Remote Sensing of Environment, pp. 97–131.

Lavrenko, V., Schmill, M., Lawrie, D., Ogilvie, P., Jensen, D., Allen, J., 2000. Mining of Concurrent Text and Time Series. In: Proc. of the 6th Int. Conference on Knowledge Discovery and Data Mining, pp. 37–44.

Li, J., 1996. A Polyalgorithm for Parallel Dense Matrix Multiplication on two-dimensional process grid topologies. Thesis: Missisippi State University.

Li, C., Yu, P., Castelli, V., 1998. MALM: A framework for mining sequence database at multiple abstraction levels. In: Proc. of the 9th Int. Conference on Information and Knowledge Management, pp. 267–272.

Lu, D., Mausel, P., Brondizio, E., Moran, E., 2004. Change detection techniques. Int. J. Remote Sens. 25 (37), 2365–2401.

Moisen, G.G., Meyer, M.C., Schroeder, T.A., Liao, X., Schleeweis, K.G., Freeman, E.A., Toney, C., 2016. Shape selection in Landsat time series: a tool for monitoring forest dynamics. Global Change Biol. 22 (10), 3518–3528.

Mougel, P.N., Folcher, N.S., 2012. A data mining approach to discover collections of homogeneous regions in satellite image time series. In: Geoscience and Remote Sensing Symposium (IGARSS), pp. 4360–4363.

Neilsen, A., Conradsen, K., Simpson, J., 1998. Multivariate alteration detection (MAD) and MAF post processing in multi-spectral bi-temporal image data: new approaches to change detection studies. Remote Sens. Environ. 64, 1–19.

Petitjean, F., Gancarski, P., Masseglia, F., Forestier, G., 2010. Analysing satellite image time series by means of pattern mining. Lect. Notes Comput. Sci. 6283, 45–52.

Petitjean, F., Inglada, J., Gancarski, P., 2011. Satellite image time series analysis under time warping. IEEE Trans. Geosci. Remote Sens. 50 (8), 3081–3095.

Petitjean, F., Kurtz, C., and Gancarski, P., 2012. Spatio-Temporal Reasoning for the Classification of Satellite Image Time Series. In: Pattern Recognition Letters, 14 pages.

Ramer, U., 1972. An iterative procedure for the polygonal approximation of planar curves. Comput. Graph. Image Process. 1, 244–256.

Rice, J.R., 1967. On the construction of polyalgorithms for automatic numerical analysis. Interact. Syst. Exp. Appl. Math. 301–313.

Rice, J.R., 1969. A Polyalgorithm for the Automatic Solution of Nonlinear Equations. In: Proc. of the 1969 24th National Conference, pp. 179–183.
Rice, J.R., 2014. Numerical Methods in Software and Analysis. Elselvier.
Rice, J.R., Rosen, S., 1966. NAPSS – a numerical analysis problem solving system. In: Proc. of the ACM National Conference, pp. 51–56.
Richards, J.A., 1984. Multitemporal analysis of Landsat imagery for monitoring forest cutovers in Nova Scotia. Canad. J. Remote Sens. 11, 188–194.
Rudin, L., Osher, S., Fatemi, E., 1992. Nonlinear total variation based noise removal algorithms. Phys. D: Nonlinear Phenom. 60, 259–268.
Saxena, R., Watson, L.T., Thomas, V.A., Wynne, R.H., 2017a. Scaling constituent algorithms of a trend and change detection polyalgorithm. In: Proc. High Performance Computing Symp. (HPC 2017), 2017 Spring Simulation Multiconference, Soc. for Modelling and Simulation Internat., Vista, CA, 12 pages.
Saxena, R., Watson, L.T., Wynne, R.H., Thomas, V.A., 2017b. Scalability of land use monitoring codes. In: Arabnia, H.R., Grimaila, M.R., Hodson, D.D., Tinetti, F.G. (Eds.), Proc. 2017 Internat. Conf. on Scientific Computing. CSREA Press, Las Vegas, NV, pp. 3–9.
Serneels, S., Said, M., Lambin, E.F., 2001. Land-cover changes around a major East African wildlife reserve: the Mara ecosystem. Int. J. Remote Sens. 22, 3397–3420.
Shatkay, H., and Zdonik, S., 1996. Approximate queries and representations for large data sequences. In: Proc. of the 12th IEEE Int. Conference on Data Engineering, pp. 546–553.
Thomson, F., Davis, G., and Colwell, J.E., 1980. Detection and measurement of changes in the production and quality of renewable resources. USDA Forest Service Final Report 145300-4-F ERIM, Ann Arbor, MI, USA.
Tucker, C.J., 1979. Red and photographic infrared linear combinations for monitoring vegetation. Remote Sens. Environ. 8 (2), 127–150.
Verbesselt, J., Hyndman, R., Newnham, G., Culvenor, D., 2010a. Detecting trend and seasonal changes in satellite image time series. Remote Sens. Environ. 114, 106–115.
Verbesselt, J., Hyndman, R., Zeileis, A., Culvenor, D., 2010b. Phenological change detection while accounting for abrupt and gradual trends in satellite image time series. Remote Sens. Environ. 114, 2970–2980.
Vintrou, E., Desbrosse, A., Begue, A., Traore, S., Baron, C., Seen, D.L., 2012. Crop area mapping in West Africa using landscape stratification of MODIS time series and comparison with existing global land products. Int. J. Appl. Earth Observ. Geoinform. 14, 83–93.
Vintrou, E., Ienco, D., Begue, A., Tesseire, M., 2013. Data mining, a promising tool for large area cropland mapping. IEEE J. Select. Top. Appl. Earth Observ. Remote Sens. 6 (5), 2132–2138.
Vlasveld, R.Q., 2014. Temporal Segmentation using Support Vector Machines in the context of Human Activity Recognition. Utrecht University.
Whittle, P., 1951. Hypothesis Testing in Time Series Analysis. Almqvist & Wiksells Boktryckeri AB, Uppsala.
Wold, H., 1938. A Study in the Analysis of Stationary Time Series. Almqvist & Wiksell.
Woodcock, E.C., Allen, R., Anderson, M., Belward, A., Bindschadler, R., Cohen, W., Gao, F., Goward, S.N., Helder, D., Helmer, E.M., Nemani, R., Oreopoulos, R., Schott, J., Thenkabail, P., Vermonte, E., Vogelmann, J., Wulder, M., Wynne, R., 2008. Free access to Landsat imagery. Science 320, 1011.
Wozniak, M., Grana, M., Corchado, E., 2014. A survey of multiple classifier systems as hybrid systems. Inform. Fusion 16, 3–17.
Xiao, P., Zhang, X., Wang, D., Yuan, M., Feng, X., Kelly, M., 2016. Change detection of built-up land: a framework of combining pixel-based detection and object-based recognition. ISPRS J. Photogram. Remote Sens. 119, 402–414.
Xing, J., Sieber, R., Caelli, T., 2018. A scale-invariant change detection method for land use/cover change research. ISPRS J. Photogram. Remote Sens. 141, 252–264.
Zhan, X., Sohlberg, A.R., Townshend, J.R.G., DiMiceli, C., Carroll, M.L., Eastman, J.C., Hansen, M.C., DeFries, R.S., 2002. Detection of land cover changes using MODIS 250 m data. Remote Sens. Environ. 83, 336–350.
Zhe, Z., 2017. Change detection using landsat time series: a review of frequencies, preprocessing, algorithms, and applications. ISPRS J. Photogram. Remote Sens. 130, 370–384.
Zhu, Z., Woodcock, C.E., 2012. Object-based cloud and cloud shadow detection in Landsat imagery. Remote Sens. Environ. 118, 83–94.
Zhu, Z., Woodcock, C.E., 2014. Continuous change detection and classification of land cover using all available landsat data. Remote Sens. Environ. 144, 152–171.