An algorithm for unsupervised partitioning of geoscientific datasets using flexible similarity metrics

Grant W. Petty[a]

[a] *Atmospheric and Oceanic Sciences, University of Wisconsin-Madison*

ABSTRACT: A simple yet flexible and robust unsupervised classification algorithm is described for efficiently partitioning a data set into compact, non-overlapping groups or classes based on pairwise similarity. Unlike most clustering algorithms, there is no assumption that natural clusters exist in the dataset, though some clusters, when present, may be preferentially assigned to one or more classes. The method also does not require data objects to be compared within any coordinate system but rather permits the user to quantify pairwise similarity using almost any conceivable criterion. For all of the above reasons, the method lends itself to certain geoscientific applications for which conventional clustering methods are unsuited, including two non-trivial and distinctly different datasets presented as examples. The computer memory required for the user-defined similarity matrix is $4N^2$ bytes and is the sole practical limitation on the size N of the dataset that can be directly classified. Much larger data sets can be readily accommodated by assigning members to classes previously determined from a representative subset.

1. Introduction

a. Background

In the Earth science disciplines in which multivariate observational data sets arise, it is sometimes desirable to group data set members into discrete, non-overlapping classes with distinct properties or interpretations. For example, the pixels in remote sensing images of the Earth's surface might be classified according to land use or type, such as open water, forest, bare ground, snow, etc., usually based on the spectral and/or textural properties determined from the image (Talukdar et al. 2020).

The assignment of classifications to the members of a data set is usually supervised in the sense that a labeled training data set with known interpretation is available to help define the criteria utilized to classify new data and to assess and refine the overall quality of the classification algorithm (Bruzzone and Demir 2014). A number of distinct methods for supervised classification exist, including artificial neural networks, classification trees, support vector machines, k-nearest neighbor, random forest, and naive Bayes (Abiodun et al. 2018; Hush and Horne 1993; Loh 2011; Maulik and Chakraborty 2017; Prasath et al. 2017; Belgiu and Drăguţ 2016; Bielza and Larranaga 2014).

Though less common, there are applications in which labeled data are unavailable and/or it is desirable to allow the data set itself to suggest specific natural groupings of data set members. In such cases, unsupervised classification is employed (Olaode et al. 2014). Most unsupervised classification schemes are based on cluster analysis, in which natural groupings of dataset members are identified from regions of higher sample density relative to the surrounding multidimensional space (Duran and Odell 2013; Kaufman and Rousseeuw 2009). Researchers have utilized cluster analysis to redefine climate zones by identifying natural groupings of multivariate climate records (Fovell and Fovell 1993; Unal et al. 2003) among other Earth sciences applications.

Common cluster analysis methods include connectivity-based or hierarchical clustering (Murtagh and Contreras 2012), centroid-based clustering (e.g, k-means; Kanungo et al. 2002)), density-based clustering (e.g., DBSCAN; Kriegel et al. 2011), and others. In general, unsupervised cluster analysis requires the user to specify parameters governing the clustering algorithm, such as the expected number of clusters and/or the target radius of the cluster in observation space.

A central underlying assumption in cluster analysis methods, of course, is that multiple semi-distinct clusters may be present in the data, and that it is the job of the algorithm to find them. Depending on the application, this may be an inappropriate assumption. Most cluster analysis methods also expect to group members of a dataset based on some measure of their proximity to one another in a multidimensional coordinate space. However, as will be discussed below, it may sometimes be desirable to define "similarity" in a more general way—one that does not make reference to a shared coordinate space.

For both reasons, there are applications for which conventional cluster analysis is ill-suited. Instead, one might require a method for efficiently *partitioning* a dataset into self-similar groups irrespective of the existence of distinct density modes and perhaps to do so without reference to an external coordinate system within which dataset members are located.

Corresponding author: Grant W. Petty, gwpetty@wisc.edu

b. Motivation

Petty and Li (2013) (hereafter PL) required a static gridded global land classification based on similarities in the climatological background microwave brightness temperature variations measured by the Tropical Rainfall Measuring Mission (TRMM) Microwave Imager (TMI; Kummerow et al. 1998). They computed long-term means and covariances of the 7-channel brightness temperatures, after excluding scenes containing precipitation. The purpose of the classification was to partition the entire land surface of the Earth into a small set of discrete classes within each of which the mean and covariance of all observations combined was as similar as possible to the means and covariances within each geographic grid box assigned to the class. The class-wide mean and covariance would then serve as a reasonable approximation to the local geophysical noise term that must be accounted for in the multichannel retrieval of light or frozen precipitation from individual observations.

Standard clustering techniques are not well suited to this problem for the following reasons:

- Mean background brightness temperatures $\mathbf{T}$ resolved on the relatively coarse geographic grid utilized by PL typically represent an admixture of highly variable land surface properties and are therefore distributed on a continuum in observation space, with no guarantee that "natural" clusters will emerge in the form of distinct density modes.
- Classification based on noise characteristics requires one to take into account not only the observed mean $\mathbf{T}$ but also the observed covariances $\mathbf{\Sigma}_T$ for each grid box. The latter are usually far from diagonal and are strongly scene-dependent. For example, a grid box that contains mixture of land and water will exhibit a very different covariance than one with the same mean brightness temperature but that is affected by variable amounts of snow and ice over the course of a year or longer. It is the combination of $\mathbf{T}$ and $\mathbf{\Sigma}_T$ that determines the degree of statistical overlap in background brightness temperatures for two different grid cells and thus the practical degree of similarity. The required calculation does not lend itself to a representation of dataset members as points in a coordinate space as required by most clustering methods.

PL therefore devised and utilized, but did not describe in detail, a heuristic unsupervised classification algorithm to create a static global map of empirical land classes at 1° resolution in latitude and longitude. To the author's knowledge, it does not resemble any other classification algorithm in wide use. Indeed it is best described as an unsupervised *partitioning* algorithm, as it does not explicitly seek to identify density modes but rather to efficiently subdivide the entire dataset into manageable number of self-similar classes based on a user-specified similarity threshold. The method is notable for its conceptual and computational simplicity, relative efficiency, and ability to accommodate an arbitrary, externally-defined metric of pairwise "similarity."

c. Overview

In the following section, the partitioning algorithm itself is outlined. Section 3 demonstrates the application of the algorithm to two simple mock data sets intended to facilitate understanding of the key properties of the algorithm. Section 4 briefly describes its application to two real but very dissimilar geoscientific data sets, the first being an update of the problem described by PL. The second data set consists of a time series of 74 years of 6-hourly meteorological analysis maps. Section 5 offers a summary and conclusions.

2. Classification Algorithm

a. Similarity function

A prerequisite for—but separate from—the classification scheme itself is an arbitrary user-defined function that computes the "similarity" between the ith and jth objects in the data set, denoted here as d_i and d_j. Specifically,

$$0 \le f(\mathrm{d}_i, \mathrm{d}_j) \le 1,$$

where $f(\mathrm{d}_i, \mathrm{d}_i) = f(\mathrm{d}_j, \mathrm{d}_j) \equiv 1$, and $f(\mathrm{d}_i, \mathrm{d}_j) = f(\mathrm{d}_j, \mathrm{d}_i) < 1$ for any pair of members that is not to be considered identical. Any pair of objects such that $f(\mathrm{d}_i, \mathrm{d}_j) = 0$ is considered to be perfectly dissimilar.

For example, a problem that requires partitioning of data based on conventional Euclidean distance between data set members might choose

$$f(\mathrm{d}_i, \mathrm{d}_j) = \frac{1}{1 - \|\mathbf{x}_i - \mathbf{x}_j\|}, \tag{1}$$

where $\mathbf{x}_i$ is the position of the ith data point in a conventional multidimensional coordinate system. There are other possible functions that accomplish the same thing; the only requirement is that the function mononotically increase from 0 to 1 as the Euclidean distance decreases to zero. Equation (1) is utilized in the first mock data example in Section 3a.

But other definitions are possible. For example, if one attribute of the data object d_i happens to be a unit vector $\hat{\mathbf{u}}_i$ representing the local orientation of a vector field, then one might define the degree of similarity in orientation as

$$f(\mathrm{d}_i, \mathrm{d}_j) = \frac{\hat{\mathbf{u}}_i \cdot \hat{\mathbf{u}}_j + 1}{2}, \tag{2}$$

```python
def classifyP22(simmat, thresh):
''' simmat:  a square, symmetric Numpy array representing the pairwise
             similarities between data set members
    thresh:  the similarity threshold to use in the classification '''

    import numpy as np

    N = simmat.shape[0]

    # Initialize output lists
    classPrototypes = []
    classMembers = []
    classAssigned = np.zeros(N).astype('int16')

    # Track original indices of rows and columns
    indexMap = np.arange(N).astype('int')

    # Initialize variables used in iteration
    unclassified = N
    classno = 0
    mask = (simmat >= thresh).astype('int8')

    # begin classification - one pass per class found
    while unclassified > 0:
        cp = np.argmax(mask.sum(axis=1))      # Find prototype for new class
        classPrototypes.append(indexMap[cp])  # Save index number for prototype
        m,  = np.nonzero(mask[cp])            # Find members of the new class
        unclassified -= len(m)
        members = indexMap[m]
        classMembers.append(members)          # Save list of members
        classno += 1
        classAssigned[members] = classno      # Update map of class assignments
        mask = np.delete(mask, m, axis=0)     # Eliminate already assigned members
        mask = np.delete(mask, m, axis=1)
        indexMap = np.delete(indexMap, m)

    return classAssigned, classPrototypes, classMembers
```

FIG. 1. A minimal working Python implementation of the basic classification algorithm. It requires the similarity matrix $\mathbf{S}$ (`simmat`)and user-specified similarity threshold T (`thresh`) as inputs.

If one were, say, a magnetostratigraphist, one might then objectively segment the magnetic field in rocks into discrete regions based on approximate local direction of the field. Equation (2) is utilized in the second mock data example in Section 3b.

For the application of PL involving n-channel microwave brightness temperature variability in geographic grid cells, the relevant similarity metric is far more complicated. It is defined in terms of the degree of overlap between two n-variate Gaussian distributions characterized by vector means $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ and covariances $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\Sigma}_j$. It is given by

$$f(\mathrm{d}_i,\mathrm{d}_j) = \exp\left(\frac{\mathbf{b}^T\mathbf{A}\mathbf{b}-c}{4}\right)\left[\frac{2^n|\mathbf{A}|}{|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}|\boldsymbol{\Sigma}_j|^{\frac{1}{2}}}\right]^{\frac{1}{2}}, \qquad (3)$$

where

$$\begin{aligned}\mathbf{A} &= (\boldsymbol{\Sigma}_i^{-1}+\boldsymbol{\Sigma}_j^{-1})^{-1}\\ \mathbf{b} &= \boldsymbol{\Sigma}_i^{-1}\boldsymbol{\mu}_i+\boldsymbol{\Sigma}_j^{-1}\boldsymbol{\mu}_j\\ c &= \boldsymbol{\mu}_i^T\boldsymbol{\Sigma}_i^{-1}\boldsymbol{\mu}_i+\boldsymbol{\mu}_j^T\boldsymbol{\Sigma}_j^{-1}\boldsymbol{\mu}_j.\end{aligned}$$

This similarity metric is utilized Section 4a.

Finally, the fourth example in Section 4b utilizes the Pearson correlation coefficient to group 500 hPa height maps, in which case the similarity function is simply

$$f(\mathrm{d}_i,\mathrm{d}_j) = \frac{\mathrm{Corr}(\mathrm{d_i},\mathrm{d_j})+1}{2}. \qquad (4)$$

b. Similarity matrix

Given a data set with N members, the primary input to the classification algorithm is a *similarity matrix* $\mathbf{S}$, the i,jth element of which is $s_{ij} = f(\mathrm{d}_i,\mathrm{d}_j)$. It is thus an $N\times N$ symmetric matrix whose diagonal elements $s_{ii}=1$ and whose off-diagonal elements $0 \le s_{ij} \le 1$.

The ability to store $\mathbf{S}$ entirely in core memory for efficient access is the most important practical limitation on the size of N that can be accommodated. As 32-bit floating point precision is more than sufficient for the elements of $\mathbf{S}$, the maximum memory requirement for $\mathbf{S}$ is approximately $4N^2$ bytes, or about 37 GB for $N = 10^5$. For significantly larger data sets, it will be shown that subsampling can be utilized to obtain initial classes, with the remaining data then being efficiently assigned to those classes.

Once $\mathbf{S}$ is available, the algorithm requires a single user-specified parameters: a similarity threshold $0 < T < 1$ that

controls the acceptable degree of dissimilarity of members occupying any class determined in the first pass. An initial $N \times N$ boolean similarity matrix is then defined as

$$\mathbf{B}_0 = \mathbf{S} > T,$$

where for computational purposes False=0 and True=1. If 8-bit integers are used, then the memory requirement of $\mathbf{B}_0$ is one-fourth that of $\mathbf{S}$.

c. The algorithm

Given the boolean matrix $\mathbf{B}_0$, the first iteration of the algorithm then proceeds as follows:

1. Each row of $\mathbf{B}_0$ is summed. The row with the largest sum determines the *prototype member* of the first class, as it is the member that is similar (to within the specified threshold T) to the most other members of the data set.

2. The columns in the selected row that are non-zero (True) determine the *members* of the class.

3. All rows and columns of $\mathbf{B}_0$ corresponding to the above members are *deleted*, leading to a reduced matrix $\mathbf{B}_1$ whose dimensions are $(N - n_1) \times (N - n_1)$, where n_1 is the number of members that were assigned to the first class in the previous step.

The cycle repeats to obtain successive class prototypes and class members and updated boolean matrix $\mathbf{B}_i$. The iteration terminates when all dataset members have been assigned to a class.

A minimal working implementation in Python is shown in Fig. 1. The algorithm requires remarkably few lines of code owing to the vectorization capabilities of the Numpy library.

d. Algorithm properties

Notable properties of the above algorithm include the following:

- The first class consists of all data set members that satisfy the similarity constraint relative to the first prototype, which in turn represents a centroid of the class, though not necessarily in a Euclidean sense. The first class is always associated with the most densely populated portion of the data space as measured within the effective radius determined by the threshold T.

- Subsequent classes are successively smaller and consist of members satisfying the similarity constraint *after* all previously assigned members of the data set have been removed from further consideration. Therefore, when a member is similar to more than than one class prototype, it is assigned to the earlier class. For the same reason, subsequent classes might or might not be associated with local density maxima. As previously noted, the goal of the algorithm is to efficiently *partition* the data set into self-similar classes, not necessarily to find distinct modes.

- The determination of k classes requires k passes through the cycle desribed in the previous subsection.

- Because the first iteration requires operations on the largest matrix $\mathbf{B}_0$, it also requires the most computational time. With each subsequent iteration, $\mathbf{B}_i$ is smaller and requires computation in proportion to the reduced array size N_i^2. Typically, the total execution time is dominated by the first one or two iterations.

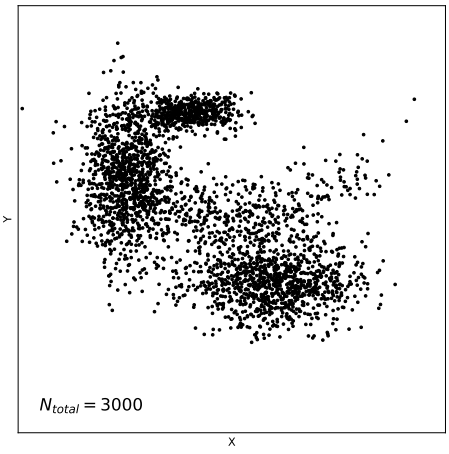

FIG. 2. Mock data used in the demonstration of classification based on Euclidean distance.

To summarize, the algorithm returns an ordered series of classes of non-increasing size, with the last of these commonly containing very few members or even just one. The disposition of those outlier classes—whether to discard them or merge them with the nearest larger class—is a decision made by the user depending on the needs of the application and is discussed below.

3. Application to Mock Data Sets

a. Example classification based on Euclidean distance

To aid in visualizing algorithm behavior, the first demonstration uses Euclidean distance (1) as the similarity metric applied to a fictitious two-dimensional data set consisting of $N = 3000$ points depicted in Fig. 2. The threshold T was chosen by trial and error to yield a total of 20 classes.

(i) Initial pass Fig. 3 depicts the results of the first nine iterations, with black dots marking the prototype member for each class as it is found. The first three classes immediately find the three major modes in the distribution of points, with the remaining classes filling in between and

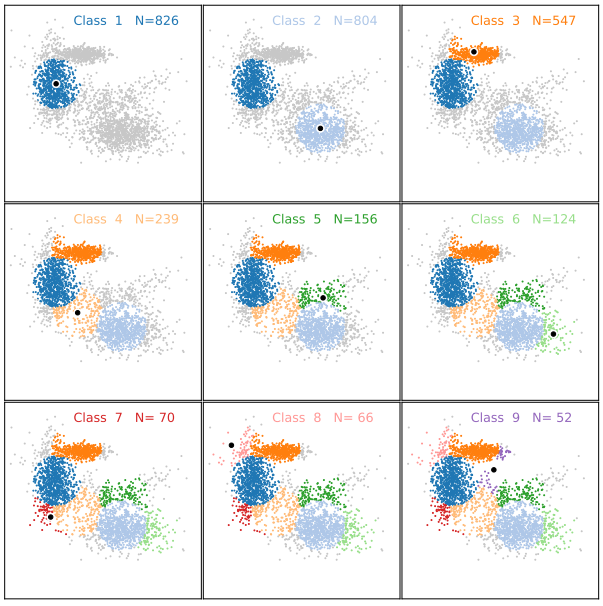

FIG. 3. Identification of the first nine classes based on a Euclidean distance threshold applied to the data in Fig. 2. Black dots indicate the prototype member defining each class.

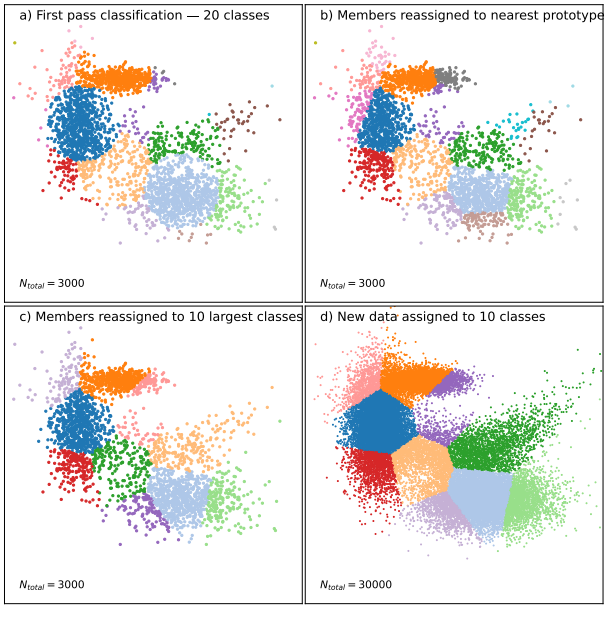

FIG. 4. The results of subsequent operations on the classification results for the data in Fig. 2. a) The initial fully classified data set. b) The classification following reassignment (reconsolidation) to the nearest class prototype. c) Reduction of classes from 20 to 10 by reassigning data in the smallest classes to the nearest larger class. d) Posterior assignment of a larger data set to the 10 classes in (c).

capturing the points on the edges of the distribution. By the end of the ninth pass, only 116 points of the original 3000 remain unclassified. These are captured by 11 additional classes (not shown), with the last four containing only one member each.

(ii) Optional reconsolidation It is possible that the initially defined classes (e.g., Fig. 4a) are satisfactory, and no additional action is needed. But we usually prefer to use the prototypes determined in the first pass to reassign all dataset members, based on the nearest (or most similar) prototype. This is a simple, efficient operation based on a reduction of the original matrix $\mathbf{S}$ to a $N \times M$ matrix $\mathbf{S}'$, where M is the number of classes found, and the columns correspond to the class prototypes. It is only necessary to find, for each row, the column corresponding the maximum similarity. This is accomplished with just one line of code using the Python/Numpy `argmax()` function. The results of this reconsolidation are shown in Fig. 4b. As long as the complete set of original prototypes is retained, the new classes are guaranteed to satisfy the original similarity threshold T. After reassignment, classes might or might not still be ordered according to descending size, so sorting and relabeling classes by size is an optional additional step.

(iii) Disposition of outliers The value of having 20 classes for 3000 points may be questionable when the first 10 classes, say, account for 98% of the dataset. An important consideration with application of this algorithm to almost any natural data set—as with many other clustering algorithms—is what to do with "outlier" classes—i.e., those containing very few members. Possibilities include the following:

1. Discard the associated data elements as unwanted outliers.

2. Retain even the smallest classes, perhaps in order to single them out as interesting examples of "rare" phenomena (see Section 4b).

3. Truncate the valid list of classes to the largest k and reassign all members of the discarded classes to a single catch-all $(k+1)$th class.

4. Truncate the list of classes to the largest k and reassign the members of discarded classes to the retained classes based on the nearest (most similar) prototype.. Note that that the expanded classes will no longer be sharply deliminated by the prescribed similarity threshold T, as seen in Fig. 4c.

The option chosen will depend on the needs of the application. For example, in PL's application discussed in Section 4a, it is necessary that all land grid cells remain in play, which eliminates option 1. It is also necessary that classes contain enough grid cells to occupy a meaningful land area in order to permit adequate satellite sampling, which eliminates option 2. Finally, the purpose of PL's classification is to ensure that noise statistics for each grid cell are well captured by the class mean and covariance,

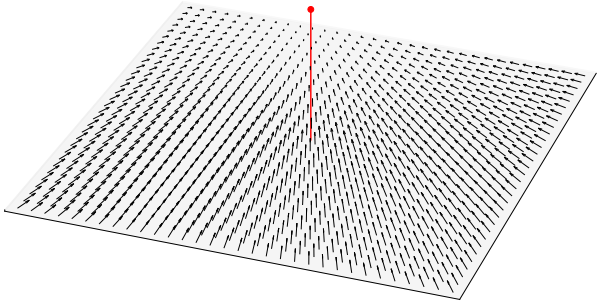

FIG. 5. Field of data elements consisting of unit vectors pointing at the red dot positioned above the plane.

which would likely not be the case for the catch-all class. This leaves option 4 as the optimal choice for their application.

(iv) Expanding the dataset size As previously noted, computer memory constraints typically become significant as N approaches $\sim 10^5$. Much larger datasets are easily accommodated by first classifying a representative subset and then assigning the remaining N_{full} dataset members based on similarity to the class prototypes. For example, if we choose to retain $k = 10$ classes, then the memory requirement for the $N_{\text{full}} \times k$ similarity matrix is just $40N_{\text{full}}$ bytes, or about 37 GB for $N_{\text{full}} = 10^9$. Much larger datasets still could be accommodated with the help of memory-mapped arrays.

If we consider the data set in Fig. 2 to be a random subset of a larger data set with $N_{\text{full}} = 3 \times 10^4$, then the resulting assignments can be seen in Fig. 4d. With resorting of the classes based on the new class sizes, some relabeling occurs, leading to the change of plotted color seen for some classes.

b. Classification based on generalized similarity

The next example is intended to illustrate the ability of the algorithm to classify data elements based on properties other than relative positions in a coordinate space. It is also a case for which there are no distinct modes of the type that would be sought out by a traditional clustering algorithm. For this purpose, we construct a 160×160 grid of elements ($N = 25\,600$) in a 2-dimensional plane, each of which is associated with a unit vector. Each vector is directed toward a common point depicted as a red dot in Fig. 5.

Using the similarity criterion given by (2), the results for one value of T are shown in Fig. 6a. For the chosen threshold, the first four classes (dark blue, light blue, dark orange, light orange) capture most of the data elements, leaving relatively few for the remaining classes to pick up. Also, there is no direct competition among the four; they all wind up with the same number of members. In this example, there

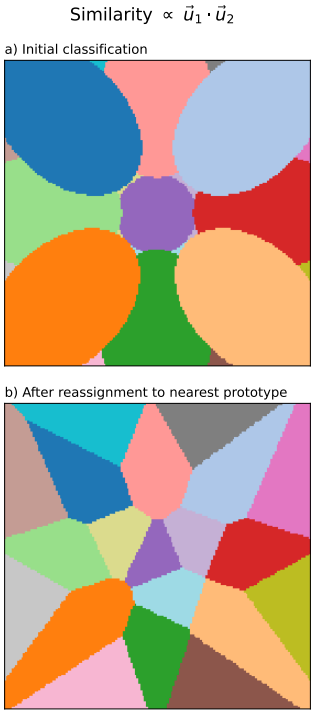

FIG. 6. Results of the classification of the grid elements in Fig. 5 according to the similarity criterion (2). a) Initial classes. b) Classes after reconsolidation.

is no significance to the initial ordering of classes, and the highly unequal distribution of classes sizes beyond the first four isn't necessarily meaningful either. This illustrates particularly well the desirability in many cases of reconsolidating classes as described earlier, producing the final classifications shown in Fig. 6b. With all members now being assigned to the "nearest" prototype, the numerical distribution among classes is now more equitable, with the initially much smaller classes having claimed "territory" back from the larger classes.

4. Applications to Earth Sciences Problems

a. Land surface classification based on microwave covariances

Here we adapt the procedure utilized by PL for TRMM to the newer Global Precipitation Measurement (GPM) Microwave Imager (GMI) 2014 (Hou et al. 2014). The starting point consists of precipitation-free microwave brightness temperatures for nine channels (10, 19, 23, 36, and 89 GHz in two polarizations) accumulated over the six-year period from 1 June 2014 through 31 May 2020. Multichannel "pseudoemissivities" were computed for each satellite pixel, defined as the brightness temperatures divided by the local surface skin temperature as reported by the ERA5 Reanalysis (Hersbach et al. 2020).

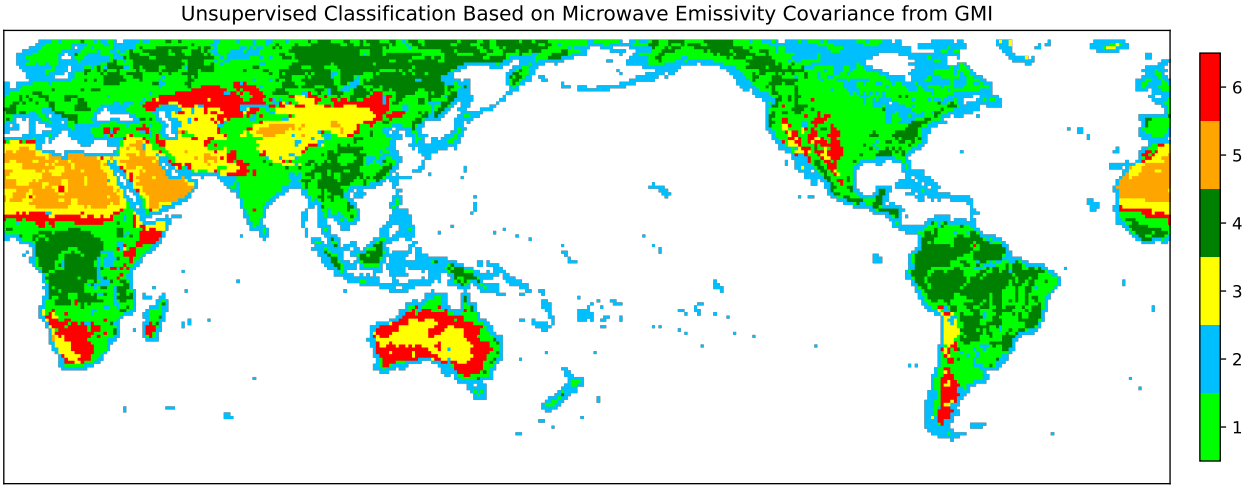

FIG. 7. Results of the classification of geographic grid boxes using similarity criterion (3) applied to means and covariances of multichannel microwave pseudoemissivity.

From the nine-channel pseudoemissivities, a single global mean and covariance was first computed for the combined land area. The first three eigenvectors of the covariance matrix, accounting for over 98% of the total variance, were then utilized to transform the individual observations from nine to only three dimenions. For each $1^\circ \times 1^\circ$ latitude-longitude grid cell containing land, means and 3×3 covariances were constructed from the transformed data.

After excluding ocean-only grid cells and polar regions not covered by the satellite swath, the warm-season data set consisted of $N = 64,800$ grid cells requiring classification. Additional smaller data sets were constructed for transitional ($N = 9,799$) and cold-season ($N = 7,055$) grid-cells, defined according to skin-temperature thresholds of 278 K and 268 K, respectively. Only the warm-season results are discussed here.

The similarity metric (3) was used to compute the matrix $\mathbf{S}$. By trial and error, a threshold $T = 0.55$ was found to yield useful groupings of land areas in the early (largest) classes, though the initial pass yielded 45 distinct classes, some with only a single member. The six largest classes were retained, and all remaining grid boxes were reassigned based on the most similar class prototype.

The result of the above procedure is depicted in Fig. 7. The six classes visibly capture geographically meaningful distinctions. Class 1, the largest class, encompasses most moderately vegetated land areas (e.g., savannah and agricultural lands) on every continent. Class 2 is clearly identified with grid cells containing significant fractions of open or standing water, including all coastlines as well as some wetter interior portions of northern Canada and Asia. Class 3 is found in the arid interiors of Australia and Asia, as well as on the margins of the African and Arabian deserts. Class 4 appears to be associated primarily with heavily forested areas, not only the rain forests of African, South American, southeast Asia, and Indonesia, but also the extensive temperate and boreal forests of North American and Asia. Class 5 is found primarily in the Saharan and Arabian deserts as well as the largely sand dune-covered Taklimakan Desert in western China. Finally, Class 6 is mostly found at the transitions from Class 1 (moderate vegetation cover) to Class 3 (arid).

It must be emphasized that the physical descriptions offered above are unimportant for the purposes of this classification and need not be validated or defended. The sole practical value in this particular exercises lies in the grouping of land areas according to their *multichannel microwave brightness temperature variability* for the purpose of characterizing the noise background for precipitation retrievals. Nevertheless, it is reassuring that this rather esoteric concern appears to lead to classifications that are readily associated with known geographic characteristics.

b. Clustering of Meteorological Analyses

As an example of an entirely different geophysical application, we turn our attention to the unsupervised classification of weather maps. For the sake of illustration, we utilize 500 hPa height fields over the continental United States from the National Center for Environmental Prediction (NCEP) Reanalysis 1 (Kalnay et al. 1996), covering the period 1 January 1948 through 31 December 2021 at 6-hourly intervals. The sample thus consists of $N = 108,116$ gridded maps of geopotential height. The maps are low-pass filtered (smoothed) to reduce the importance of small-scale features in the comparisons. The goal is to assign classes that group similar maps together.

There are different measures of similarity that could be employed, and they lead to different results. One possi-

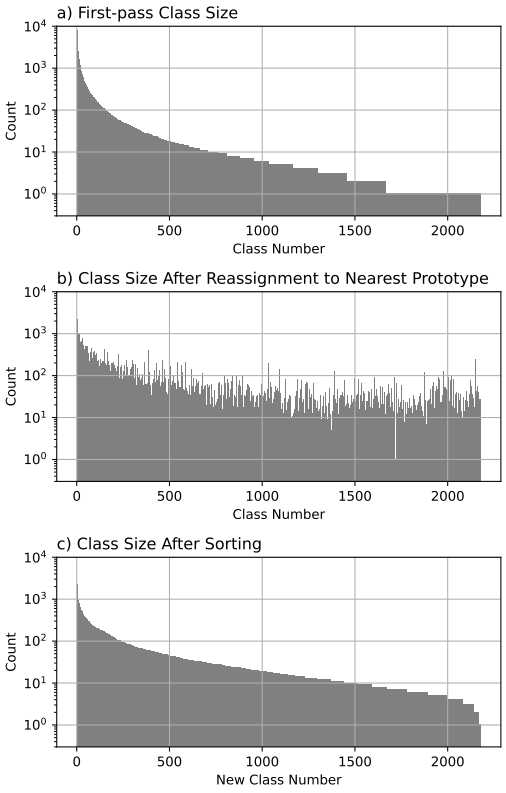

FIG. 8. Class size vs. class number resulting from application of the classification algorithm to 500 hPa height maps from the NCEP Reanalysis. a) Initial results. b) Following reconsolidation. c) Following sorting by class size.

bility is the root-mean-squared difference between gridded maps, which favors matches for which both the amplitudes and shapes of the height fields are similar. It is also the traditional Euclidean distance in a high-dimensional space. For this demonstration, we chose instead to use the Pearson correlation coefficient, and thus (4) as the similarity metric, because the $N \times N$ correlation matrix can be computed in a single operation using the efficient Python/Numpy `corrcoef()` routine.

The matrix $\mathbf{S}$ in this case was near the upper limit of what could be accommodated by the 128 GB of RAM on our desktop workstation. Nevertheless, the algorithm completed the classification in about 20 minutes, using a threshold T that corresponded to a correlation coefficient $r = 0.992$ according to (4).

The initial pass resulted in 2,178 prototypes and associated classes ranging in size from 8,114 members to just one, with almost 500 classes in the latter category (Fig. 8a). Members were then reconsolidated as usual, leading to substantial changes in the membership of each class (Fig. 8b), including many additions to previously small classes. Finally, the modified classes were sorted by size, yielding the distribution showing in Fig. 8c, with Class 1 now containing only 2,197 members, and only the final 13 classes containing one member.

The prototypes for Classes 1 through 6, encompassing the most common height patterns in the dataset and accounting for 7.9% of the total, are depicted in Fig. 9. We thus observe that the most commonly occurring patterns exhibit north-south height gradients generally consistent with the time of year but without pronounced wave structure.

Figure 10 allows us to examine six of the 13 single-member classes. Each of these is notable in being "dissimilar" ($r < 0.992$) from every other map in the dataset. An interesting potential application of the classification algorithm lies in its ability to easily find not only maps that are analogs of one another but also to single out cases for study that for whatever reason are extremely rare.

Finally, Fig. 11 depicts all members of a single randomly chosen six-member class. Interestingly, it does not consist of six independent occurrences of the same height pattern scattered over 74 years but rather a single episode that persisted over at least 30 consecutive hours. Thus, the occurrence of the pattern in question is as rare as that of any single-member class but is, unlike the others, apparently prone to some degree of persistence.

The point here is not to undertake a in-depth meteorological analysis of the classification results but rather to illustrate the potential utility of the algorithm for objectively identifying common and rare weather patterns for further study. Whether examining 500 hPa heights over North America or some other variable and some other region, one could easily examine whether certain patterns of interest have occurred with greater or lesser frequency during a selected period.

5. Conclusions

This paper presented a remarkably simple yet flexible and robust unsupervised classification algorithm for efficiently partitioning a multivariate data set into compact, non-overlapping groups or classes based on mutual similarity. It does not resemble any other classification or clustering algorithm known to the author.

Unlike clustering algorithms, the algorithm does not assume that there are "natural" clusters—i.e., multiple density modes—present in the dataset. It also does not require data objects to be referenced to any coordinate system, as required for example for the determination of Euclidean distance. Rather, the user has the freedom to define pairwise similarity arbitarily, subject only to the basic rules outlined at the beginning of Section 2. As long as those rules are satisfied, it does not appear to be possible for the algorithm to fail, and it requires just one iteration per class found, starting with the largest.

The sole significant practical limitation is the computer memory required for the $N \times N$ similarity matrix, which

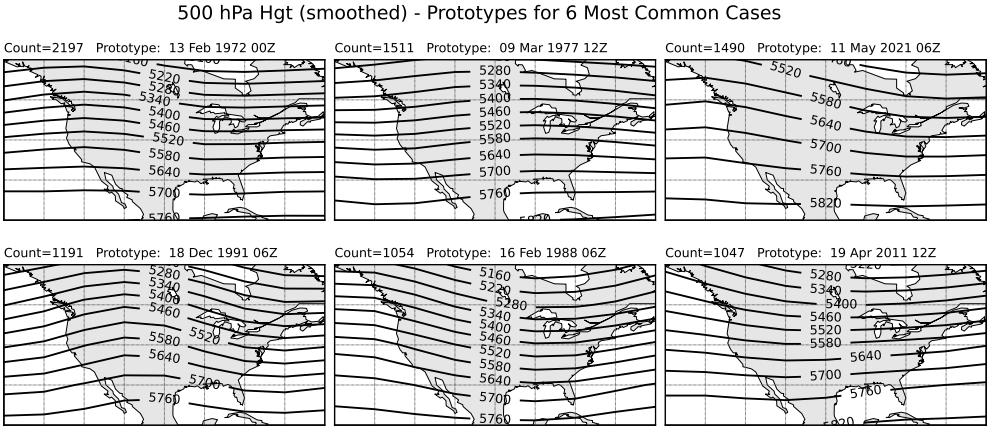

FIG. 9. Prototypes of the most-populous six classes of 500 hPa height patterns identified in the 74-year NCEP Reanalysis record.

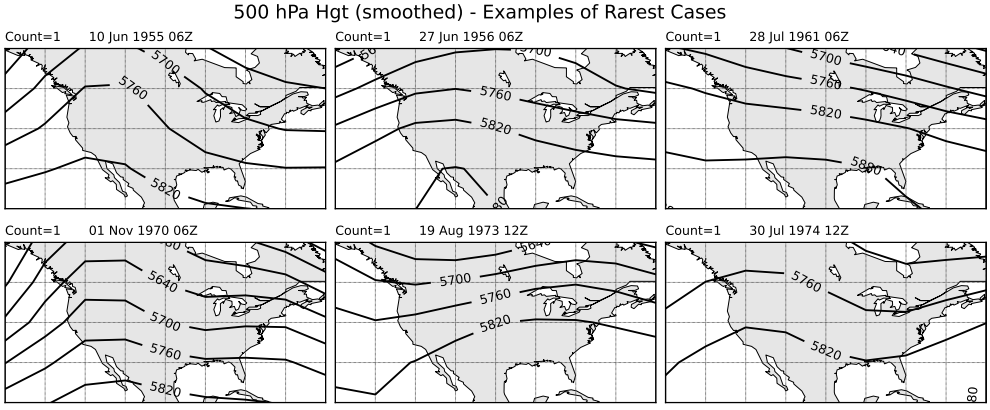

FIG. 10. Six of the 13 single-member classes of 500 hPa height patterns identified in the 74-year NCEP Reanalysis record.

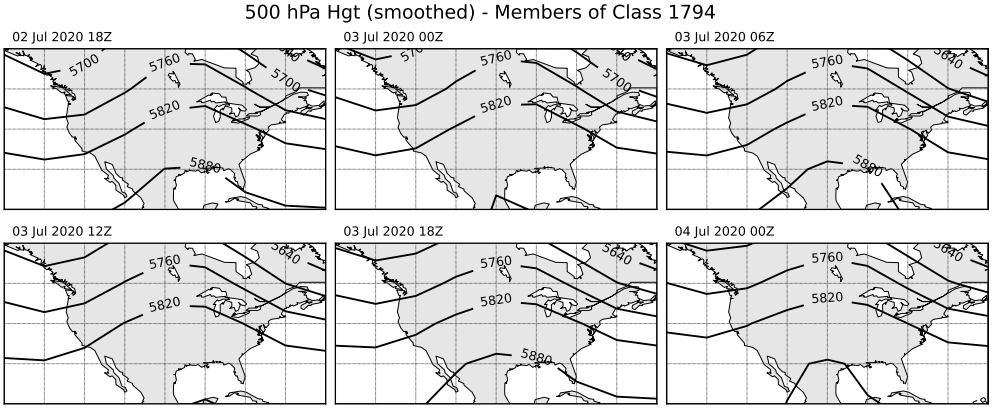

FIG. 11. All six members of a single arbitrarily selected six-member class of 500 hPa height patterns identified in the 74-year NCEP Reanalysis record.

begins to pose a problem for many desktop computers as the dataset size approaches 10^5. But much larger data sets can easily be accommodated by retroactively assigning new members to classes previously determined from a representative subset, based on the most similar class prototype.

It was shown that the algorithm could be meaningfully applied to four very different data sets, including two nontrivial geoscientific datasets, using very different similarity metrics. It was also demonstrated that a simple reconsolidation of classes following the initial classification usually leads to a more meaningful distribution of classes.

When applied to statistics of microwave multichannel brightness temperature variability obtained from extended satellite observations of the Earth, the resulting classifications bore obvious relationships to known regions of desert, forest, coastlines, etc. And when applied to a 74-year record of gridded weather data, the method readily identified both very common and very rare patterns of 500 hPa height fields.

Acknowledgments. The author thanks Jerry (Xiaojin) Zhu for helpful comments on the relationship of the method described herein to other classification methods. This work was partially supported by NASA Grant NNX16AF70G through the Precipitation Measurement Mission (PMM).

Data availability statement. The author's full Python code implementing the algorithm described herein is available from `github.com/gpetty/classification`. Datasets and Jupyter notebooks used in the examples may be requested from the author.

References

Abiodun, O. I., A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, 2018: State-of-the-art in artificial neural network applications: A survey. *Heliyon*, **4 (11)**, e00 938.

Belgiu, M., and L. Drăguţ, 2016: Random forest in remote sensing: A review of applications and future directions. *ISPRS journal of photogrammetry and remote sensing*, **114**, 24–31.

Bielza, C., and P. Larranaga, 2014: Discrete Bayesian network classifiers: A survey. *ACM Computing Surveys (CSUR)*, **47 (1)**, 1–43.

Bruzzone, L., and B. Demir, 2014: A review of modern approaches to classification of remote sensing data. *Land Use and Land Cover Mapping in Europe*, 127–143.

Duran, B. S., and P. L. Odell, 2013: *Cluster analysis: a survey*, Vol. 100. Springer Science & Business Media.

Fovell, R. G., and M.-Y. C. Fovell, 1993: Climate zones of the conterminous United States defined using cluster analysis. *Journal of climate*, **6 (11)**, 2103–2135.

Hersbach, H., and Coauthors, 2020: The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, **146 (730)**, 1999–2049.

Hou, A. Y., and Coauthors, 2014: The Global Precipitation Measurement mission. *Bulletin of the American Meteorological Society*, **95 (5)**, 701–722.

Hush, D. R., and B. G. Horne, 1993: Progress in supervised neural networks. *IEEE signal processing magazine*, **10 (1)**, 8–39.

Kalnay, E., M. Kanamitsu, R. Kistler, W. Collins, D. Deaven, L. Gandin, and Coauthors, 1996: The NCEP/NCAR 40-year reanalysis project. *Bulletin of the American Meteorological Society*, **77 (3)**, 437–471.

Kanungo, T., D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, 2002: An efficient k-means clustering algorithm: Analysis and implementation. *IEEE transactions on pattern analysis and machine intelligence*, **24 (7)**, 881–892.

Kaufman, L., and P. J. Rousseeuw, 2009: *Finding groups in data: an introduction to cluster analysis*, Vol. 344. John Wiley & Sons.

Kriegel, H.-P., P. Kröger, J. Sander, and A. Zimek, 2011: Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **1 (3)**, 231–240.

Kummerow, C., W. Barnes, T. Kozu, J. Shiue, and J. Simpson, 1998: The tropical rainfall measuring mission (TRMM) sensor package. *J. Atmos. Ocean. Tech.*, **15 (3)**, 809–817.

Loh, W.-Y., 2011: Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, **1 (1)**, 14–23.

Maulik, U., and D. Chakraborty, 2017: Remote sensing image classification: A survey of support-vector-machine-based advanced techniques. *IEEE Geoscience and Remote Sensing Magazine*, **5 (1)**, 33–52.

Murtagh, F., and P. Contreras, 2012: Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **2 (1)**, 86–97.

Olaode, A., G. Naghdy, and C. Todd, 2014: Unsupervised classification of images: a review. *International Journal of Image Processing*, **8 (5)**, 325–342.

Petty, G. W., and K. Li, 2013: Improved passive microwave retrievals of rain rate over land and ocean. Part I: Algorithm description. *J. Atmos. Ocean. Tech.*, **30 (11)**, 2493–2508.

Prasath, V., H. A. A. Alfeilat, A. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, and H. S. E. Salman, 2017: Distance and similarity measures effect on the performance of k-nearest neighbor classifier–a review. *arXiv preprint arXiv:1708.04321*.

Talukdar, S., P. Singha, S. Mahato, S. Pal, Y.-A. Liou, A. Rahman, and Coauthors, 2020: Land-use land-cover classification by machine learning classifiers for satellite observations—a review. *Remote Sensing*, **12 (7)**, 1135.

Unal, Y., T. Kindap, and M. Karaca, 2003: Redefining the climate zones of Turkey using cluster analysis. *International Journal of Climatology: A Journal of the Royal Meteorological Society*, **23 (9)**, 1045–1055.