# The Pairwise Similarity Partitioning algorithm: a method for unsupervised partitioning of geoscientific and other datasets using arbitrary similarity metrics

Grant W. Petty[a]

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

ABSTRACT: A simple yet flexible and robust algorithm is described for fully partitioning an arbitrary dataset into compact, non-overlapping groups or classes, sorted by size, based entirely on a pairwise similarity matrix and a user-specified similarity threshold. Unlike many clustering algorithms, there is no assumption that natural clusters exist in the dataset, though 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 define pairwise similarity using almost any conceivable criterion. The method therefore lends itself to certain geoscientific applications for which conventional clustering methods are unsuited, including two non-trivial and distinctly different datasets presented as examples. In addition to identifying large classes containing numerous similar dataset members, it is also well-suited for isolating rare or anomalous members of a dataset. The method is inductive, in that prototypes identified in representative subset of a larger dataset can be used to classify the remainder.

## 1. Introduction

### *a. Background*

In the Earth science disciplines in which multivariate observational datasets arise, it is sometimes desirable to group dataset 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 dataset is usually supervised in the sense that a labeled training dataset 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 dataset itself to suggest natural groupings of dataset members. In such cases, unsupervised classification is employed (Olaode et al. 2014). Most unsupervised classification schemes are based on cluster analysis, in which the intent is usually to identify natural groupings of dataset members 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. Unsupervised cluster analysis generally requires the user to specify one or more parameters governing the clustering algorithm, such as the expected number of clusters and/or the target radius of the cluster in observation space. A number of the most widely used algorithms are described and compared, with demonstration code, in the online Scikit-learn documentation (Pedregosa et al. 2011; Scikit-learn 2022).

There are situations in which the most common assumptions of clustering algorithm do not hold. First, it may be desirable to partition a dataset in a manner that does not necessarily presume the existence of distinct density maxima. Second, it may be necessary to employ a similarity metric other than distance between points in a coordinate space or any of the other standard metrics available as options in some clustering libraries. For both reasons, there are applications for which the mostly widely used clustering algorithsm are ill-suited.

### *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; Kum-

*Corresponding author*: Grant W. Petty, gwpetty@wisc.edu merow et al. 1998). They computed spatially gridded one-year 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 $1^\circ$ resolution in latitude and longitude. To the author's knowledge, it does not resemble any other clustering or classification algorithm in wide use. Its goal is to efficiently subdivide the entire dataset into self-similar classes based on a user-specified similarity threshold that is enforced on the members of every class. The method is notable for its conceptual and computational simplicity and ability to accommodate an arbitrary, externally-defined metric of pairwise similarity.

### c. *Overview*

In the following section, the partitioning algorithm itself, henceforth referred to as the Pairwise Similarity Partitioning (PSP) algorithm, is described. Section 3 demonstrates the application of PSP to two mock datasets intended to facilitate understanding of the key properties of the algorithm. Section 4 demonstrates its application to two dissimilar geoscientific datasets. The first is an update of the problem undertaken by PL, as described above. The second consists of a time series of 74 years of 6-hourly meteorological analysis maps. Section 5 offers a summary and conclusions.

## 2. The Algorithm

### a. *General characteristics*

As previously noted, a wide variety of clustering and partitioning algorithms exist, many of which have become standard tools in the data sciences. Before proceeding with the computational details of the PSP algorithm, a few key characteristics may be highlighted to facilitate comparisons with existing methods. In particular,

- It is a strict partitioning algorithm, with each dataset member being assigned to one and only one class.

- It is centroid-based, choosing specific dataset members to serve as "prototypes" for a particular class.

- Classes are found in order of decreasing size, with the first class capturing the most dataset members and the final classes typically capturing solitary outliers.

- It is inductive: the prototypes can be used to classify additional members not seen during the initial analysis.

- It has a single parameter—a similarity threshold—that determines the minimum similarity of members in each class relative to that class's prototype. The same parameter thus also indirectly controls the total number the classes obtained.

- It is deterministic: presented with the same dataset, in any order, it will find the same initial groupings for any particular value of the similarity threshold.

- Unlike most clustering algorithm, it finds the first complete class on the first pass through the algorithm loop. Subsequent passes address the as-yet unclassified members. It is thus iterative only in the sense that one pass is required for each class found; no optimization or gradient descent procedure is involved. Iteration continues until the last dataset member is classified.

- Unlike the case for most partitioning algorithms, the pairwise similarity metric employed is external to the algorithm and may be defined by the user according to the task at hand.

While some other clustering algorithms share some of the above features, none is known to the author to share all of them. The flexibility of the similarity metric and the enforcement of a known similarity threshold on all resulting classes are among the most important distinguishing features of the PSP algorithm. Depending on the problem considered, these same features can make it difficult to undertake apples-to-apples comparisons with other partitioning algorithms.

### *b. Similarity function*

A prerequisite for—but separate from—the PSP scheme itself is an arbitrary user-defined function that computes the similarity between the $i$th and $j$th objects in the dataset, denoted here as $\mathrm{d}_i$ and $\mathrm{d}_j$. 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. Note that the formal definition of a metric additionally requires that

$$f(\mathrm{d}_i, \mathrm{d}_k) \le f(\mathrm{d}_i, \mathrm{d}_j) + f(\mathrm{d}_j, \mathrm{d}_k);$$

however, it is unclear, without further investigation, whether that condition has any practical relevance to the PSP.

For example, a problem that requires partitioning of data based on conventional Euclidean distance between dataset 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 $i$th 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 similarity functions are possible. For example, if one attribute of the data object $\mathrm{d}_i$ 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

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

Given an observed or modeled vector field, it could then be partitioned 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

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

where

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

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

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

### *c. Similarity matrix*

Given a dataset with $N$ members, the primary input to the classification algorithm is the similarity matrix $\mathbf{S}$, the $i,j$th 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 random-access memory (RAM) 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 $N = 10^5$. In certain cases, one might be able to reduce the storage requirement by up to a factor of four by replacing floating point values with unsigned 1-byte or 2-byte integers and scaling the threshold range discussed below accordiningly. For significantly larger datasets, 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 parameter: a similarity threshold $0 < T < 1$ that controls the acceptable maximum degree of dissimilarity of members occupying any class as compared to the class's prototype. An initial $N \times N$ boolean similarity matrix is

```python
def PSP(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.

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 $\mathbf{S}$.

### *d. 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 dataset. The prototype serves as the centroid of the class, though not necessarily in a Euclidean sense.

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 useful analogy might be made to principle component analysis (PCA). In PCA, each successive eigenvector explains the maximum possible fraction of the remaining variance, subject to the orthogonality requirement. In the PSP method, each successive class captures the maximum possible number of remaining dataset members, subject to the similarity threshold.

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.

### *e. Algorithm properties*

Notable properties of the above algorithm include the following:

- The first prototype always consists of the member that is similar (to within $T$) to the most other members of the dataset.

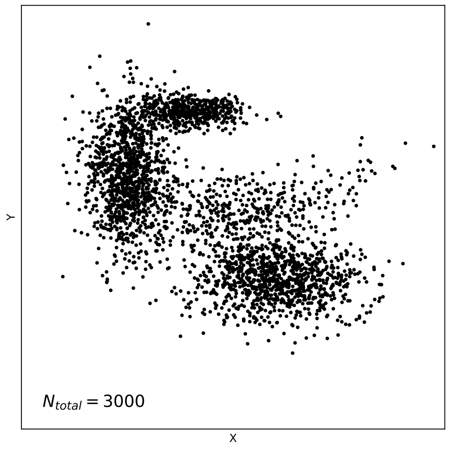

FIG. 2. A sample of the mock data used in the demonstration of classification based on Euclidean distance.

- The first class therefore always captures the most densely populated regions of the data space on the scale of $T$.
- Subsequent classes are successively smaller and consist of members satisfying the similarity constraint after all previously assigned members of the dataset 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.
- 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 earliest few iterations.

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 Datasets

### *a. Example classification based on Euclidean distance*

To aid in visualizing algorithm behavior, the first demonstration of PSP uses Euclidean distance (1) as the similarity metric applied to a randomly generated two-dimensional

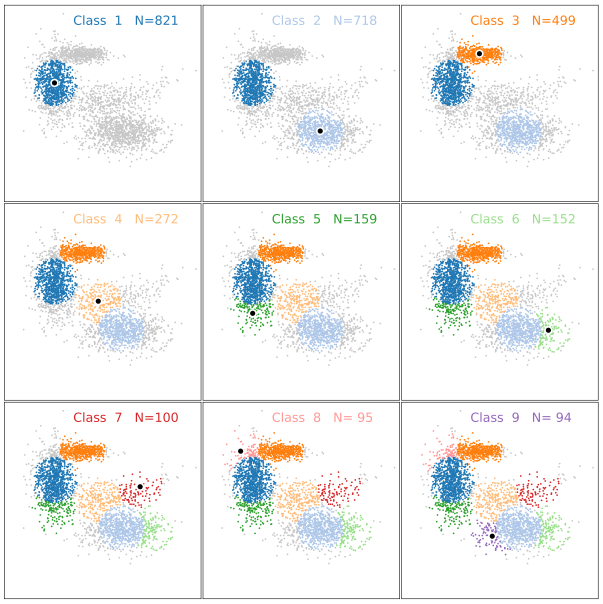

FIG. 3. Identification of the first nine classes based on a Euclidean distance threshold applied to the data in Fig. 2. Black dots indicate the prototype member defining each class.

dataset. A representative subset, consisting of $N = 3000$ points, is depicted in Fig. 2. For the initial classification of this subset, the similarity threshold $T$ was chosen to be 0.3.

#### 1) 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 capturing the points on the edges of the distribution. By the end of the ninth pass, only 90 points, or 3%, of the original 3000 remain unclassified. These are captured by 13 additional classes for a total of 22 (Fig. 4a), with the last three containing only one member each.

#### 2) OPTIONAL REASSIGNMENT

It is possible that the initially defined classes 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 then 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()`

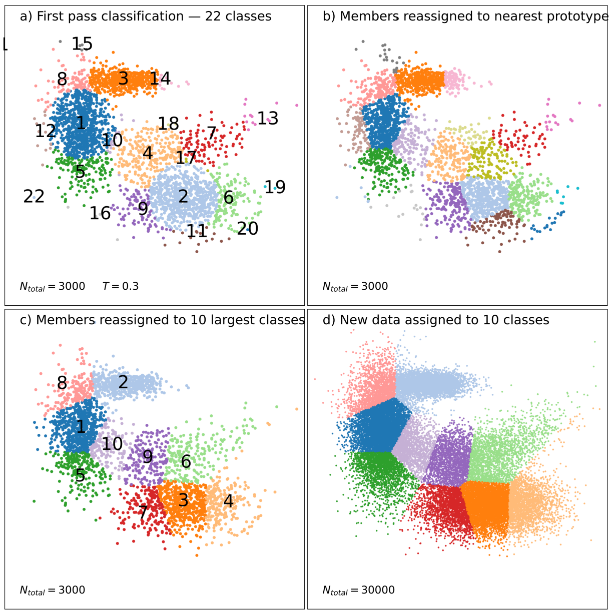

FIG. 4. The results of subsequent operations on the classification results for the data in Fig. 2. a) The initial fully classified dataset, with class numbers (ordered by size) shown. Class 21 is assigned to a single outlier and falls outside the domain of the plot. b) The classification following reassignment (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 dataset ($N = 30000$) to the 10 classes in (c).

function. The results of this reassignment 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 are usually no longer strictly ordered according to size, so sorting and relabeling classes by size is a recommended additional step.

### 3) Disposition of outliers

The value of having 22 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 dataset—as with many other clustering algorithms—is what to do with classes containing very few members. Possibilities include the following:

1. Discard the associated data elements as unwanted outliers.

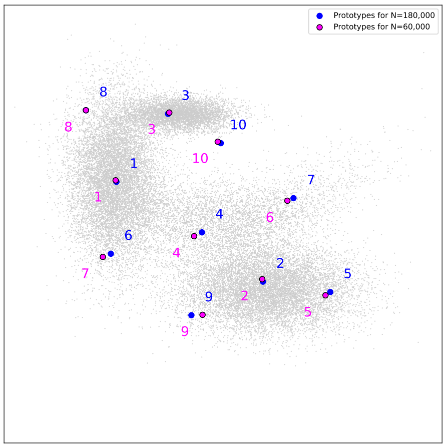

FIG. 5. Comparison of class prototypes derived from a large dataset consisting of $1.8 \times 10^5$ points (blue dots and labels) with those obtained from a random subset of $6 \times 10^4$ points (magenta).

2. Retain even the smallest classes, perhaps 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 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 essential 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, which would likely not be the case for a single catch-all class. This leaves option 4 as the optimal choice for their application.

### 4) Expanding the dataset size

As previously noted, computer memory requirements typically become problematic as $N$ approaches $\sim 10^5$. There are two ways to accommodate larger datasets:

- Careful implementation of memory-mapped arrays; e.g., using Numpy's `memmap()` function. While this in principle solves the problem of accommodating arrays larger than the available RAM on the users's computer, there is a large cost in processing time, due to the inherently slow nature of disk access.

- Initial classification based on a representative subset of the data, with the remaining data then being assigned to the nearest class prototypes.

Here we briefly demonstrate the second approach. If we consider the dataset in Fig. 2, containing $N = 3000$ points, to be a random subset of a larger dataset with $N_{\text{full}} = 3 \times 10^4$, then the resulting assignments can be seen in Fig. 4d. Later, we will consider the robustness of the class definitions when subsetting is employed.

An important question is whether classification based on subsampling leads to similar outcomes relative to direct classification of the entire dataset. To examine this issue, a dataset consisting of $1.8 \times 10^5$ points—a statistically identical superset of the data already considered in this section—was processed on a desktop Linux server with 128 GB of RAM and just able to accommodate the 121 GB similarity matrix $\mathbf{S}$. Memory-mapped arrays were implemented to help manage intermediate results when calculating $\mathbf{S}$. The creation of $\mathbf{S}$ itself required approximately 1.5 hours of wall clock time. The classification itself, executed entirely in RAM, required just over an hour. The first 10 class prototypes are depicted in Fig. 5 as blue dots.

The same procedure, but without the use of memory-mapped arrays was conducted on a subset consisting of $6 \times 10^4$ points, requiring 13.4 GB for $\mathbf{S}$ and thus potentially within the reach of common laptop computers equipped with only 16 GB of RAM. The computation of $\mathbf{S}$ required 23 s, and the classification required 15 s. The resulting prototypes are shown as magenta dots in Fig. 5. We see that the prototypes obtained from the subset are very close to those from the full dataset; thus assignment of the full dataset to prototypes derived from the subset would lead to similar but not identifical classifications, with only points near the boundaries between classes being affected. Classes 6 and 7 are seen to reverse order, but only because these were already of similar size.

## 5) Varying the similarity threshold

The PSP algorithm has only one adjustable parameter, the similarity threshold $T$. The number—and thus granularity—of classes can be adjusted at will by varying this threshold. For $T = 0$, all data points are encompassed by a single class. For $T \to 1$, every point becomes the sole member of its own class, and the number of identified classes equals the number of points. For less extreme variations in $T$, the effect is illustrated in Fig. 6.

Regardless of the choice of $T$, there are usually outliers in a dataset that initially get assigned to very small or even single-member classes. The total number of classes resulting from a first-pass classification is thus an unreliable guide. While outlier classes can be eliminated in principle by sufficiently reducing $T$, this almost always results in the earlier classes becoming too large to be useful. $T$ should therefore be chosen first to optimize the distinctions between the earlier major classes. Truncation and, if desired, reassignment can then be used to eliminate unwanted outlier classes.

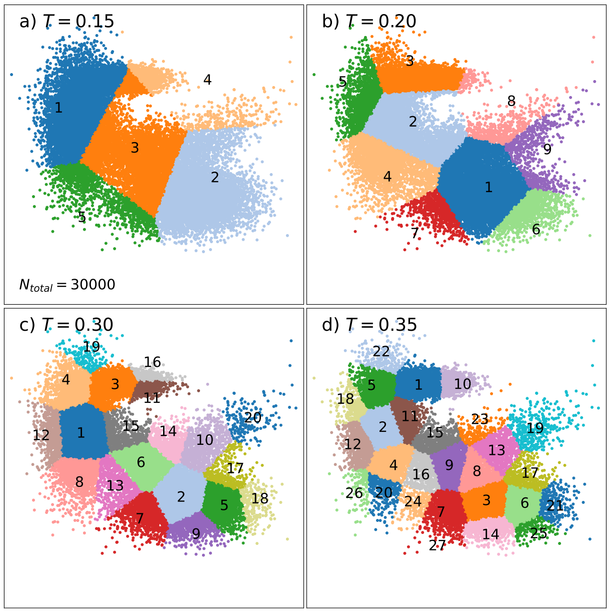

Fig. 6. The effect of varying the similarity threshold $T$, shown here after reassignment as described in Section 3a2.

Thus, unlike some other machine learning schemes having multiple parameters, there is only a one-dimensional space that needs to be explored to find a value of $T$ that comes closest to achieving the user's objective. For any dataset whose precomputed $N \times N$ similarity matrix $\mathbf{S}$ can be accommodated by the user's computer, it typically takes only seconds to compute and examine the results of each value of $T$. Trial and error is therefore an viable strategy for choosing $T$.

## 6) Comparison with k-means

In this first demonstration of the PSP algorithm, we are using the traditional Euclidean distance for the similarity metric. It therefore offers an opportunity to compare the PSP results with those from the more familiar k-means clustering algorithm, whose results for 10 clusters are shown in Fig. 7. The k-means partitioning of the dataset is qualitatively similar to that obtained from PSP, as seen in Fig. 4d. Nevertheless, several differences may be noted, some of which may be important depending on the application. These include the following:

- Unlike PSP, k-means does not sequentially determine classes in order of maximum points encompassed but

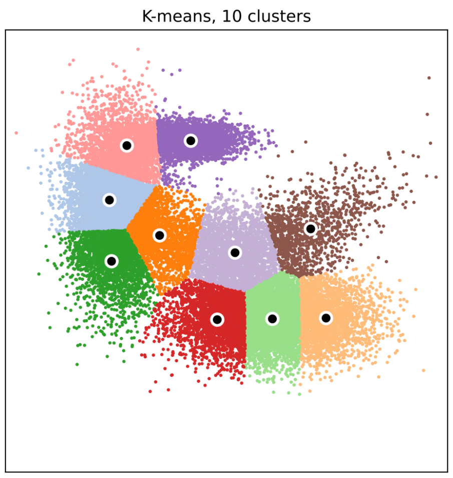

FIG. 7. The k-means algorithm applied to the same mock dataset as in Fig. 4d. Dots indicated the cluster centroids.

rather globally optimizes all centroids to minimize the sum of the squared distances from the centroids. Centroids and their corresponding classes thus have no intrinsic ranking by importance. They may however be sorted by size as in Section 4b.

- The centroids obtained from k-means do not correpond to specific members of the dataset, so the method does not directly allow one to examine a 'prototype" member of the class. One would instead need to search for the member closest to any given centroid.

- k-means is less prone to creating classes or clusters that straddle gaps or thin spots in the data distribution, as seen for PSP in Fig. 6b for classes 8 and 9, for example. This is because k-means penalizes distance from the centroid, whereas PSP is sensitive only to the number of points within a bounding radius in similarity space.

- Unlike PSP, which is deterministic, k-means begins with a random initialization of centroids and then iteratively optimizes those centroids. The final result depends on the initialization, and one typically conducts several runs and selects the result with the best mean-squared error.

- With k-means, one cannot impose strict constraints on the maximum acceptable distance between points within a cluster. The effective radius may vary substantially between clusters.

- Due to the above characteristic, k-means also does not assign distinct clusters to outlier cases. This is a disadvantage if one is specifically interested in anomaly detection (see Section 4b).

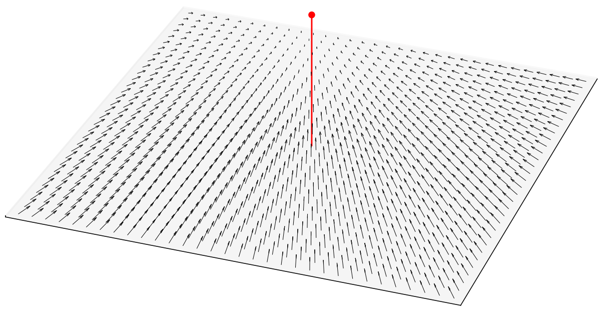

FIG. 8. Field of data elements consisting of unit vectors pointing at the red dot positioned above the plane.

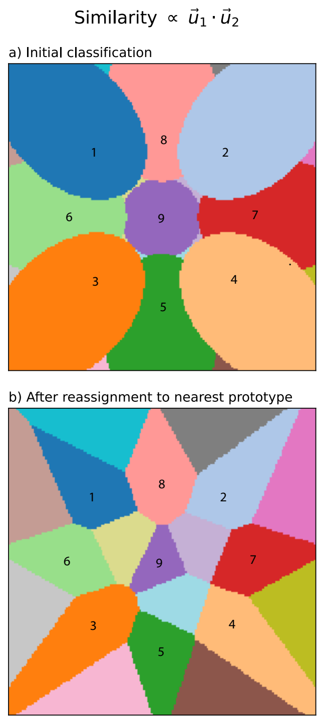

FIG. 9. Results of the classification of the grid elements in Fig. 8 according to the similarity criterion (2). Only the first nine classes of 29 are numbered. a) Initial classes. b) Classes after reassignment.

All such differences are moot when the goal is to classify a dataset based on something other than Euclidean distance. There are of course other clustering algorithms besides k-means, including those documented and compared by Scikit-learn (2022). Those that can accommodate arbitrary non-Euclidean similarity metrics—e.g., Agglomerative Clustering—have other major differences from PSP with respect to both objective and generality.

### *b. Classification based on relative orientation*

The second mock dataset illustrates the ability of the PSP algorithm to classify data elements based on properties

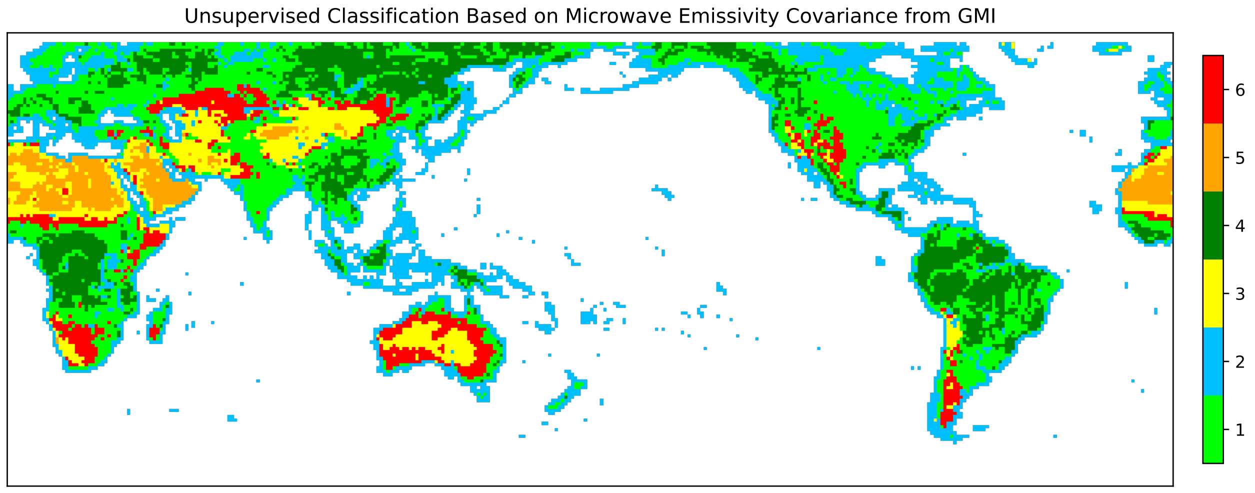
Unsupervised Classification Based on Microwave Emissivity Covariance from GMI
6
5
4
3
2
1

Fig. 10. Results of the classification of geographic grid boxes using similarity criterion (3) applied to means and covariances of multichannel microwave pseudoemissivity.

other than relative positions in a coordinate space. It is also a case for which the data field is homogeneous—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. 8.

Using the similarity criterion given by (2), the results for $T = 0.93$ are shown in Fig. 9a. The first four classes capture a large share of the members. There is no competition between them, and due to the symmetry of the setup, all four are of exactly equal size. Classes 5–9 then capture most of the remainder, leaving a few for the remaining classes 10–29 (not labeled) to pick up. In this example, there is no significance to the precise ordering of classes, and the highly unequal distribution of classes sizes beyond the first eight is potentially misleading. This case illustrates especially well the desirability of reassigning classes as described earlier, producing the final classifications shown in Fig. 9b. 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 reclaimed "territory" back from the larger classes.

## 4. Applications to Earth Sciences Problems

We now turn to demonstrations of the PSP algorithm applied to non-trivial Earth Sciences datasets. It is beyond the scope of this paper to elaborate on the scientific contexts or significance of the specific applications. The focus here is strictly on illustrating the behavior and potential uses of the PSP method.

### *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 intent is facilitate the discrimination of the passive microwave signature of falling precipitation from background brightness temperatures that vary strongly in both time and space (Petty 2013). Related but methodologically distinct efforts are described by Aires et al. (2011) and Turk et al. (2021).

#### 1) Setup

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).

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 dimensions. For each $1^\circ \times 1^\circ$ latitude-longitude grid cell containing land, means and $3 \times 3$ 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 dataset consisted of $N = 64,800$ grid cells requiring classification. Additional smaller datasets 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 **S**, which required 15.6 GB of RAM. Because of the unusually complex similarity calculation, it took ∼1 hour to compute this modest-sized array.

As discussed in Section 3a, trial and error is efficient enough for subjectively choosing an optimal 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 as usual based on the most similar class prototype.

### 2) Results

The result of the above procedure is depicted in Fig. 10. 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 rainforests of Africa, South America, 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 exercise 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 esoteric concern appears to lead to classifications that are readily associated with known geographic characteristics.

## b. Clustering of Weather Maps

As an example of an entirely different geophysical application, we turn our attention to the unsupervised classification or grouping of weather maps. There is a long history of attempts to forecast weather by finding past weather patterns—analogues—that match current patterns and therefore provide insight into future developments.

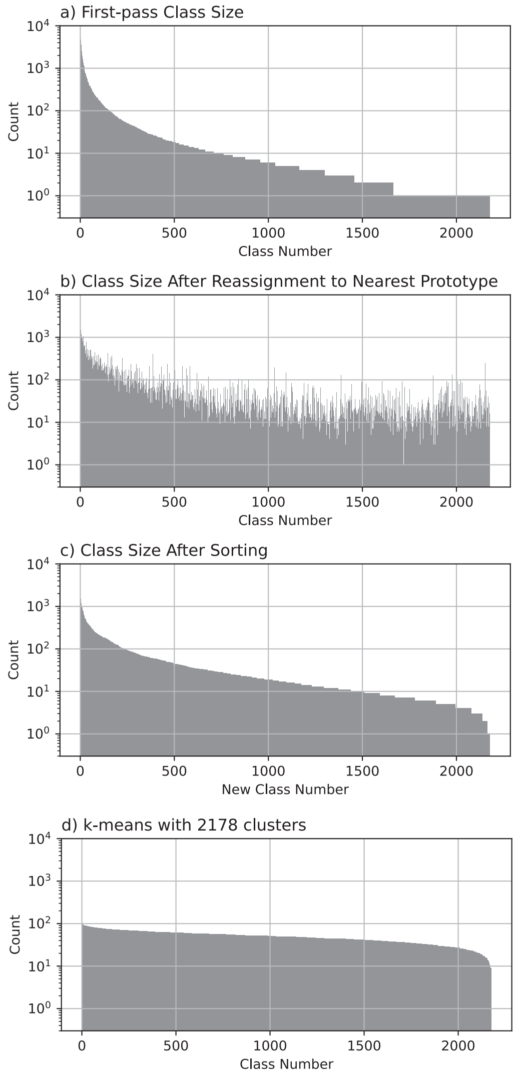

Fig. 11. Class size vs. class number resulting from application of the PSP algorithm to 500 hPa height maps from the NCEP Reanalysis. a) Initial results. b) Following reconsolidation. c) Following sorting by class size. d) Comparison with k-means clustering for the same number of classes.

The analogue method mostly failed to live up to expectations, in large part because of the rarity of sufficiently accurate matches in the historical record, though some progress has been made in overcoming this handicap (Van den Dool 1989; Hamill et al. 2015).

Here we focus on a different question: can the PSP algorithm be utilized to identify certain weather patterns that are unusually common or, perhaps more interestingly, unusually rare? The second case can be regarded as a form of anomaly detection; i.e., the identification of notable outliers in a dataset that may have special significance (Chandola et al. 2009). In other data science contexts, anomalies might be associated with erroneous or fraudulent records.

In the present context, they might represent weather patterns that are so rare as to be worthy of further study. The relevant property of the PSP algorithm is its automatic assignment of outliers to classes of their own. A data object that is the sole member of a class is by definition one that is less similar than the chosen threshold $T$ to every other object in the dataset.

#### 1) Setup

For the sake of illustration only, 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). This field is chosen because it is commonly regarded by meteorologists as one of the most representative depictions of the general state of the atmosphere in the mid-troposphere. The dataset employed here covers 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 and subsampled at $10^\circ$ latitude/longitute intervals to reduce both the data volume and the importance of small-scale features in this demonstration.

There are different measures of similarity that could be employed, and they lead to different results. One possibility 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 45-dimensional space, corresponding to the $5 \times 9$ grid size of the subsetted map arrays. For this problem, we instead used the Pearson correlation coefficient, and thus (4), so as to again demonstrate the complete flexibility one has in the choice of similarity metric.

The matrix $\mathbf{S}$ in this case required 43 GB of RAM, and 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). Since anomaly detection is one prospective goal, this threshold was chosen by trial and error to yield a small but non-zero number of single-member classes after reassignment.

#### 2) Results

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. 11a). Members were then reassigned as usual, leading to substantial changes in the membership of each class (Fig. 11b), including many additions to previously small classes. Finally, the modified classes were sorted and relabeled by size, yielding the distribution showing in Fig. 11c, with Class 1 now containing only 2,197 members, and only the final 13 classes containing one member each.

The prototypes for Classes 1 through 6, encompassing the most common height patterns in the dataset and collectively accounting for 8% of the total record, are depicted in Fig. 12. We observe that the most commonly occurring patterns exhibit north-south height gradients generally consistent with the time of year but without marked wave structure.

Figure 13 depicts a representative sample of six of the 13 single-member classes. Each of these is notable in being "dissimilar" ($r < 0.992$) from every other member in the dataset. A common feature of these exceptional cases is that the gradient in 500 hPa height is remarkably flat relative to the more common cases. From a meteorological standpoint, this implies an unusually uniform lower-atmospheric thermal structure across the region.

Finally, Fig. 14 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 across 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.

It must be emphasized that the point here is not to undertake an in-depth meteorological analysis of the classification results but rather to illustrate the potential utility of the algorithm for objectively identifying both common and rare dataset members for possible further study. Among other things, one might look at whether specific patterns occur with changing frequency over the time period covered by the record and which might be especially prone to persisting over longer periods of time.

#### 3) Comparison with k-means

The same dataset was also passed to the standard k-means algorithm with the requested number of clusters set to match that found by the PSP algorithm. Note that the clustering criterion used by k-means is the Euclidean distance in 45-dimensional space, not correlation coefficient, so we cannot exactly compare results. However, we can look at the sorted distribution of cluster sizes in Fig. 11d. The distribution is far flatter than the PSP results, with the largest cluster containing only 105 members (as compared to 2,197 for PSP after reassignment) and the smallest class containing 7. The vast majority of classes have between 30 and 90 members.

Even allowing for the different similarity metric, it is clear that k-means does not impose a reasonably predictable and invariant similarity threshold—i.e., cluster radius in Euclidean terms—across classes. As a result, we lose much of the ability to characterize the relative frequency of different patterns by reference to a reasonably constant similarity threshold. That includes the ability to isolate single outliers, which are a natural product of the

PSP algorithm. When the goal is specifically to control the minimum degree of self-similarity within clusters and/or meaningfully characterize the relative frequency of occurrence of particular patterns subject to a rigid similarity constraint, the PSP method seems preferable to k-means.

## 5. Conclusions

This paper presented a remarkably simple yet flexible and robust unsupervised classification—the Pairwise Similarity Partitioning (PSP) method—for efficiently partitioning a multivariate dataset into compact, non-overlapping groups or classes based on a user-specified threshold of mutual similarity and sorted by size. It does not resemble any other classification or clustering algorithm known to the author.

Unlike many algorithms, PSP does not necessarily assume that there are "natural" clusters—i.e., multiple distinct 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.

A significant practical limitation is the computer memory required for the $N \times N$ similarity matrix, which begins to pose a problem for many desktop computers as the dataset size approaches $10^5$. But the method is inductive, in that much larger datasets can be accommodated by retroactively assigning new members to classes previously determined from a representative subset, based on the most similar class prototype. At least in a two-dimensional domain, it was shown that the determination of class prototypes (or centroids) is relatively insensitive to training on a subset, provided that the subset is large enough to be statistically representative while still being within reach of a computer equipped with only 16 GB of RAM.

It was also shown that the algorithm could be meaningfully applied to four distinctly different datasets, including two non-trivial geoscientific datasets, using distinctly different similarity metrics. When applied to gridded statistics of microwave multichannel brightness temperature variability, the resulting classifications bore obvious relationships to known regions of desert, forest, coastlines, etc. It bears emphasizing that the unusual similarity metric (3) was both specific and essential to the context for this analysis while also being impossible to replicate using any other standard partitioning or clustering algorithm.

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. Obvious future extensions of this work include 1) examining other regions (e.g., the entire northern hemisphere) and/or meteorological variables (e.g., 850–500 hPa thickness) of interest to atmospheric dynamicists to identify common and rare patterns of possible meteorological or climatological significance; 2) looking for trends or interdecadal variations in the frequency of occurrence of specific patterns, and 3) examining the relative stability or persistence over time of a particular pattern.

Throughout this paper, the focus has been on the initial description and demonstration of what appears to be a novel data analysis tool. While the examples provide are drawn from the Earth sciences, there is no obvious reason to limit its application to any specific discipline. The sole requirement is the ability to define a meaningful metric of pairwise similarity, whether between images, hyperspectral pixels within a set of images, weather maps, atmosphere temperature and humidity profiles, or even text documents.

*Acknowledgments.* The author thanks Jerry (Xiaojin) Zhu for helpful comments on the relationship of the method to other classification methods. Three anonymous reviewers made suggestions that led to substantial improvements of this paper. The author thanks one reviewer for the suggestion of the algorithm name adopted herein. This work was partially supported by NASA Grants NNX16AF70G and 80NSSC22K0601 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.

Aires, F., C. Prigent, F. Bernardo, C. Jiménez, R. Saunders, and P. Brunel, 2011: A tool to estimate land-surface emissivities at microwave frequencies (TELSEM) for use in numerical weather prediction. *Quarterly Journal of the Royal Meteorological Society*, **137 (656)**, 690–699.

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.

Chandola, V., A. Banerjee, and V. Kumar, 2009: Anomaly detection: A survey. *ACM computing surveys (CSUR)*, **41 (3)**, 1–58.

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.

Hamill, T. M., M. Scheuerer, and G. T. Bates, 2015: Analog probabilistic precipitation forecasts using GEFS reforecasts and climatology-calibrated precipitation analyses. *Monthly Weather Review*, **143 (8)**, 3300–3309.

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.

Pedregosa, F., and Coauthors, 2011: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.

Petty, G., 2013: Dimensionality reduction in bayesian estimation algorithms. *Atmospheric Measurement Techniques*, **6 (9)**, 2267–2276.

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*.

Scikit-learn, 2022: Clustering. URL https://scikit-learn.org/stable/modules/clustering.html.

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.

Turk, F. J., and Coauthors, 2021: Adapting passive microwave-based precipitation algorithms to variable microwave land surface emissivity to improve precipitation estimation from the GPM constellation. *Journal of Hydrometeorology*, **22 (7)**, 1755–1781.

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.

Van den Dool, H., 1989: A new look at weather forecasting through analogues. *Monthly weather review*, **117 (10)**, 2230–2247.

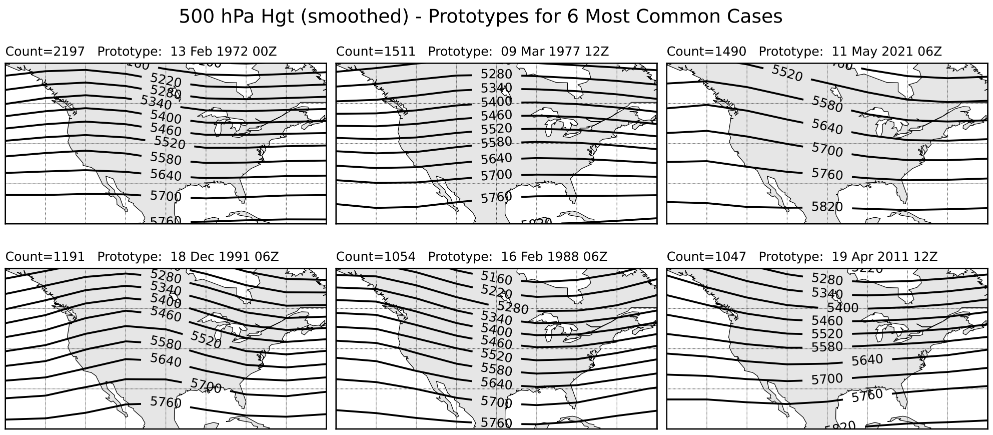
500 hPa Hgt (smoothed) - Prototypes for 6 Most Common Cases
Count=2197 Prototype: 13 Feb 1972 00Z
Count=1511 Prototype: 09 Mar 1977 12Z
Count=1490 Prototype: 11 May 2021 06Z
Count=1191 Prototype: 18 Dec 1991 06Z
Count=1054 Prototype: 16 Feb 1988 06Z
Count=1047 Prototype: 19 Apr 2011 12Z

Fig. 12. Prototypes of the most-populous six classes of 500 hPa height patterns identified in the 74-year NCEP Reanalysis record.

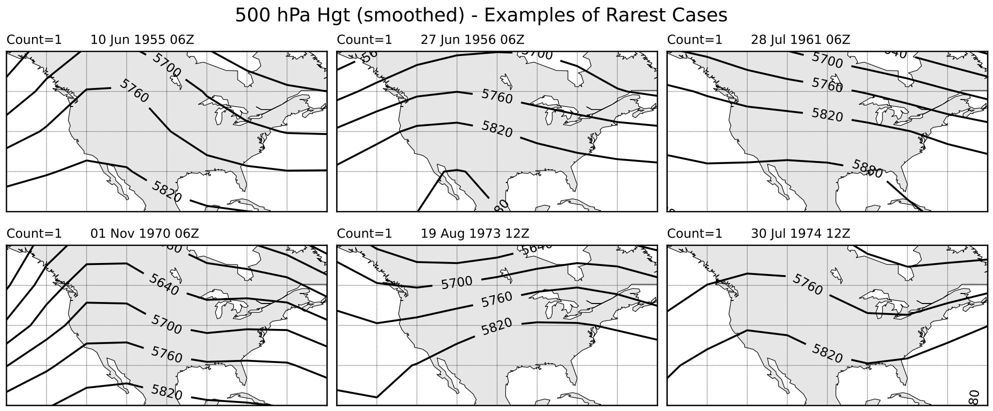

Fig. 13. Six of the 13 single-member classes of 500 hPa height patterns identified in the 74-year NCEP Reanalysis record.

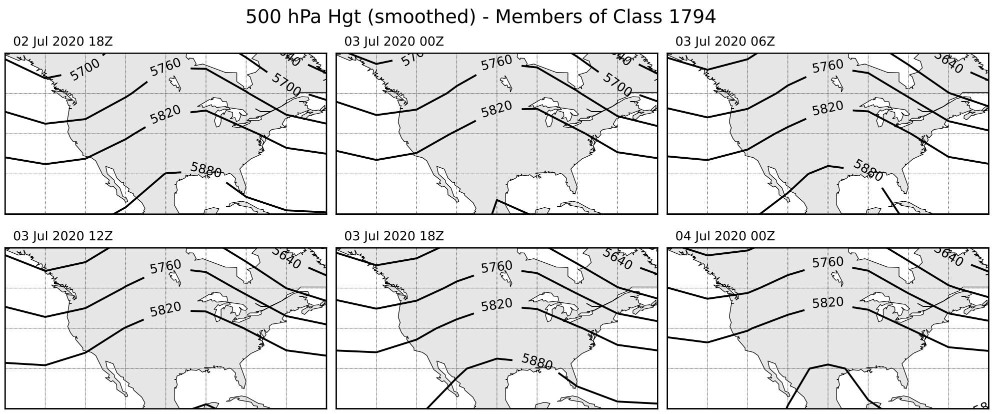

Fig. 14. All six members of a single arbitrarily selected six-member class of 500 hPa height patterns identified in the 74-year NCEP Reanalysis record.