The `growclusters` Package for R October 2022

Randall Powers*, Wendy Martinez+, and Terrance Savitsky*

*Office of Survey Methods Research, U.S. Bureau of Labor Statistics
+U.S. Census Bureau

Powers.Randall@bls.gov

Abstract: The `growclusters` package for R implements an enhanced version of *k*-means clustering that allows discovery of local clusterings or partitions for a collection of data sets that each draw their cluster means from a single, global partition. The package contains functions to estimate a partition structure for multivariate data. Estimation is performed under a penalized optimization derived from Bayesian non-parametric formulations. This paper describes some of the functions and capabilities of the `growclusters` package, including the creation of R Shiny applications designed to visually illustrate the operation and functionality of the `growclusters` package.

Disclaimer: Any opinions and conclusions expressed herein are those of the authors and do not reflect the views of the Bureau of Labor Statistics or the U.S. Census Bureau.

Key words: clustering; R Shiny; unsupervised learning; *k*-means clustering, text analysis; hierarchical clustering

1. Background

Cluster analysis is the grouping of data such that data records or observations assigned to the same group (or cluster) are more like each other than those in other groups. Clustering is a key tool in exploratory data analysis. It is an iterative process that can be achieved using a variety of algorithms and approaches [Martinez, et al., 2011], and one should try different clustering methods to search for structure or groups in the data.

The `growclusters` package for R contains functions designed to estimate a clustering or partition structure for relatively high-dimensional multivariate data. Estimation is performed under a penalized optimization derived from Bayesian non-parametric formulations [Savitsky, 2016].

Given that clustering is used to explore one's data set, creation of an interactive data visualization tool to accompany the package was a top priority and inspiration for this project. Building an R

Shiny application that would allow `growclusters` package users to visualize clustering outcomes was determined to be the best solution to achieve this goal.

This project is ongoing; currently we are working on three R Shiny applications to accompany the package. The first, presently called `gendata`, allows the user to create a customized input data set to be used in the R Shiny app that performs the clustering called `dpGrowclusters`. The third R Shiny app to be developed will implement the hierarchal version of `growclusters` and will be called `hdpGrowclusters`. For this paper, we will focus on the functionality of the latter two apps. For more information on previous work and descriptions of the applications, see Powers, et al. [2019, 2020, 2021].

To date, we have used the `growclusters` functions to analyze two data sets. One was a collection of papers from a series of workshops hosted by the United Nations Economic Commission for Europe [Martinez and Savitsky, 2019]. The second was a corpus of Bureau of Labor Statistics (BLS) *Monthly Labor Review* (MLR) articles from 2000 to 2013 [Powers, et al., 2021]. The *Monthly Labor Review* (MLR) is the principal journal of fact, analysis, and research published by the BLS. More information, including published articles, can be found at https://www.bls.gov/opub/mlr/. Articles by economists, statisticians, and other experts from BLS and stakeholders provide a wealth of knowledge on subjects pertaining to a wide range of economic issues. We use the MLR data to illustrate the functionality of the applications discussed in this paper.

2. Example Data Set

We briefly describe the MLR data set used to illustrate the functionality of the R Shiny apps developed for the `growclusters` package. The BLS makes data, articles, and resources available to the public on their website and tries to make them discoverable and accessible. The BLS Office of Publications sought to develop a taxonomy for MLR articles that could be used to categorize and tag the articles making them easier to locate. A taxonomy is defined as a system (often hierarchical) used to classify, organize, describe, and name items, which in our case are journal articles. The authors decided to use clustering or unsupervised learning applied to a sample of MLR articles as a starting point for the taxonomy, since finding clusters is a natural way to organize the articles.

The MLR articles used in this paper were published between 2000 and 2013 and should have some common topics over this time period. We could take all the articles and cluster them as if they were published at the same time, i.e., in the same year. This would be what we are calling the *single-source* approach. Note that this viewpoint is the typical way one would cluster data, and many clustering methods exist in the literature, such as *k*-means clustering [Martinez, et al., 2011]. We could also cluster articles published in each year separately, which assumes there is no dependence among topics over the years. The hierarchical `growclusters` method finds global topics just as in the single-source idea, but accounts for possible dependencies of articles between journal years. With the MLR data set, the sub-domains would be the year of publication.

We had 574 MLR articles to cluster, which were encoded (i.e., converted from text to numbers) using the bag-of-words method [Solka, 2008], after the usual pre-processing steps were performed

(e.g., remove stop words, convert to one case, etc.). Two types of encoding were used: raw frequency (the number of times a word appears in a document) and binary (1 indicates the word is present in the document and 0 means the word is absent). The dimensionality of the data set is equal to the number of unique words across all articles or documents, which is typically very high. The dimensionality (or number of features) for this data set was 12,437. This is way too many features for 574 observations, so the dimensionality was reduced using Isometric Feature Mapping (ISOMAP). Using ISOMAP, we reduced the dimensionality to 3 dimensions for the binary-encoded data and 4 dimensions for the encoding based on raw word frequencies [Martinez, et al., 2011].

3. The `dpGrowclusters` Application

The `dpGrowclusters` application allows the user to perform the clustering analysis for what we called the single-source clustering viewpoint. Recall that for that clustering context, we assume that the data do not have inherent sub-domain structures. If the user does not have a data set to work with, then they could randomly generate one using the `gendata` Shiny application we developed. The `gendata` app has been described in previous JSM papers; see [Powers, et al., 2019] for details. Note that the `gendata` app generates data corresponding to the single-source concept only at this point.

The `dpGrowclusters` application contains five tabs. When the application opens, the tab displayed is the `Welcome` tab (see Figure 1). This tab simply describes the functionality of the other four tabs and gives the user some helpful background information.

The user would next click on the `Load Data` tab (see Figure 2). They are presented with the question of where they want to load their data from. They can choose to load the data from their computer storage or from the R Studio workspace. If they choose the former, they browse their directory, select their RDS file, and the contents of the file are displayed in table format. If they choose the other option, then a list of appropriate data objects in the working space is presented in a drop-down menu. The user selects a data object, then clicks the `Show Data` button, and the file is displayed in tabular form.

Once the user has the data loaded into the app, the textbox containing the file variables to display is populated. The user can then select the desired variables for plotting and click the `Produce Matrix Plot` button. Once the button is pushed, a matrix plot is produced. At this point, no clustering has taken place. The purpose of the matrix plot is to allow the user to examine the data and visually explore the data for groupings. See Figures 3 and 4 for screenshots showing the results of loading the data from a directory (Figure 3) and the working space (Figure 4).

To cluster the data, one clicks on the `dpCluster` tab (Figure 5). Here, clustering is performed on the data that was loaded in the previous tab. The user chooses one of three methods to determine the optimal clustering parameter, which in turn influences the estimated number of clusters [Savitsky, 2016]. The three methods are: cross-validation, the silhouette statistic, and the Calinski-Harabasz statistic [Martinez, et al., 2011]. The user can also specify other inputs, including the

maximum number of clusters allowed. Once the user selects a method and runs the algorithm, a grouping or partition is produced. A bar plot shows the estimated number of clusters found. The bar heights indicate the number of observations (or documents in this case) in each cluster.

The user may then switch to the `Scatter Plot` tab (see Figure 6). The `Scatter Plot` tab shows a matrix plot where the colors indicate cluster or group membership. The user can visually explore the cluster results in this plot. The user can specify what variables to display, just as they did in the `Load Data` tab. Both this tab and the fifth tab are dependent upon clustering having been performed in the `dpCluster` tab. If the clustering has not been performed, there is nothing to display.

The fifth tab (see Figure 7) is titled the `Parallel Plot` tab. This shows the data in a parallel coordinates plot. Each broken line is one observation (or document) in the data set. Bundles of lines with similar pathways through the vertical coordinate axes indicate good clusters or that the observations are close together. As with the `Scatter Plot` tab, the colors indicate cluster membership determined from `dpCluster`. The user can also choose to highlight and view a single cluster by graying out the others (see Figure 8). Note that the colors assigned to cluster IDs are not the same in the scatter plot matrix and the parallel coordinates plot.

4. The `hdpGrowclusters` Application

In some cases, we might encounter data sets that have a known inherent group structure. For example, observations could represent papers published in a series of workshops like the UNECE data or articles published in annual volumes of a journal as was the case with the MLR data set. In the first one, the known groups or sub-domains would be the workshops, and with the MLR data, the known groups correspond to the year of publication. Another motivating application was described in the original paper by Savitsky [2016], where the known sub-domain structures were the NAICS codes assigned to establishments (https://www.census.gov/naics/).

Using the MLR articles to illustrate the concepts in the following discussion, we could ignore the known sub-domains (or year of publication) and apply clustering to the articles as if they were published at the same time. This is the single-source clustering approach described previously and implemented in the `dpGrowclusters` R Shiny application.

The single-source clustering does not account for the fact that articles were published over a period of years. Another approach might be to separately apply the single-source clustering to the articles published in each individual year. However, global topics would be difficult to merge and to label across the years. Keep in mind that we want to find global classes or groups for all MLR articles, regardless of the year, to create a common taxonomy. We would like to find global clusters for all MLR articles in the corpus *and* account for possible dependencies based on the year of publication. This is the purpose behind the original hierarchical clustering approach by Savitsky [2016]. Global clusters that account for possible local dependencies among known sub-domains are called *hierarchical* `growclusters`. This should not be confused with hierarchical clustering, which is a clustering method that has been around since at least the 1960s; see Martinez, et al. [2011]. The hierarchical `growclusters` method is inspired by Bayesian hierarchical models but is not Bayesian.

An application called `hdpGrowclusters` similar to `dpGrowclusters` is under development. There are analogous ways to obtain optimal parameters (cross-validation, Calinski-Harabasz, silhouette), which will also appear in the hierarchical version. The additional information on sub-domains gives rise to different types of visualizations to be included in the `hdpGrowclusters` app. These planned visualizations are listed below:

- The user will be able to highlight and view the observations in specified sub-domains in the `Load Data` tab.
- The scatter plot matrix showing cluster membership (similar to Figure 6) will be provided, and the user will have the option to view clusters for individual sub-domains.
- An additional tab for visualizing the results from clustering will be developed for this app. This new tab will allow the user to see bar plots showing the number of documents (or observations) conditional on either the cluster ID or the sub-domain.

Future Work and Final Comments

Our next step is to complete the `hdpGrowclusters` app, as described in the previous section. We will then write several vignettes illustrating the functionality of the `growclusters` package and its accompanying R Shiny apps. We plan to submit the package to CRAN and/or GitHub once it is ready for public release.

This paper focuses on the development of R Shiny apps and not so much on the functionality included in the `growclusters` package. The approach by Savitsky [2016] allows for clustering of data collected using a complex sample design and incorporates sampling weights. Future work will explore this additional clustering framework with additional data sets and will illustrate its properties.

5. References

Martinez, Wendy L., Angel R. Martinez, and Jeffrey L. Solka. (2011). *Exploratory Data Analysis with MATLAB*. CRC Press.

Martinez, Wendy L., and Terrance Savitsky. (2019). Towards a Taxonomy of Statistical Data Editing Methods. Working Paper presented at the United Nations Economic Commission for Europe Conference of European Statisticians. https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.44/2018/T_USA_Martinez_Paper.pdf

Powers, R. K., W. Martinez, and T. Savitsky. (2021). Creating a Data-Driven Taxonomy. 2021. In JSM Proceedings, Statistical Computing Section. Alexandria, VA: American Statistical Association. 932-945.

Powers, R. K., W. Martinez, and T. Savitsky. (2020). Integration of Two RShiny Applications into the `growclusters` Package with an Example Implementation. In JSM Proceedings, Statistical Computing Section. Alexandria, VA: American Statistical Association. 801-817.

Powers, R. K., W. Martinez, and T. Savitsky. (2019). Creation of Two R Shiny Applications to Illustrate and Accompany the `growclusters` Package. In JSM Proceedings, Statistical Computing Section. Alexandria, VA: American Statistical Association. 2468-2478.

Savitsky, Terrance D. (2019). About `growclusters`. Documentation for `growclusters` package (to be published on CRAN)

Savitsky, Terrance D. (2016). Scalable Approximate Bayesian Inference for Outlier Detection under Informative Sampling, *Journal of Machine Learning Research* 17(225):1−49.

Solka, Jeffrey L. (2008). Text Data Mining: Theory and Methods, *Statistics Surveys*. 2:94–112, https://doi.org/10.1214/07-SS016

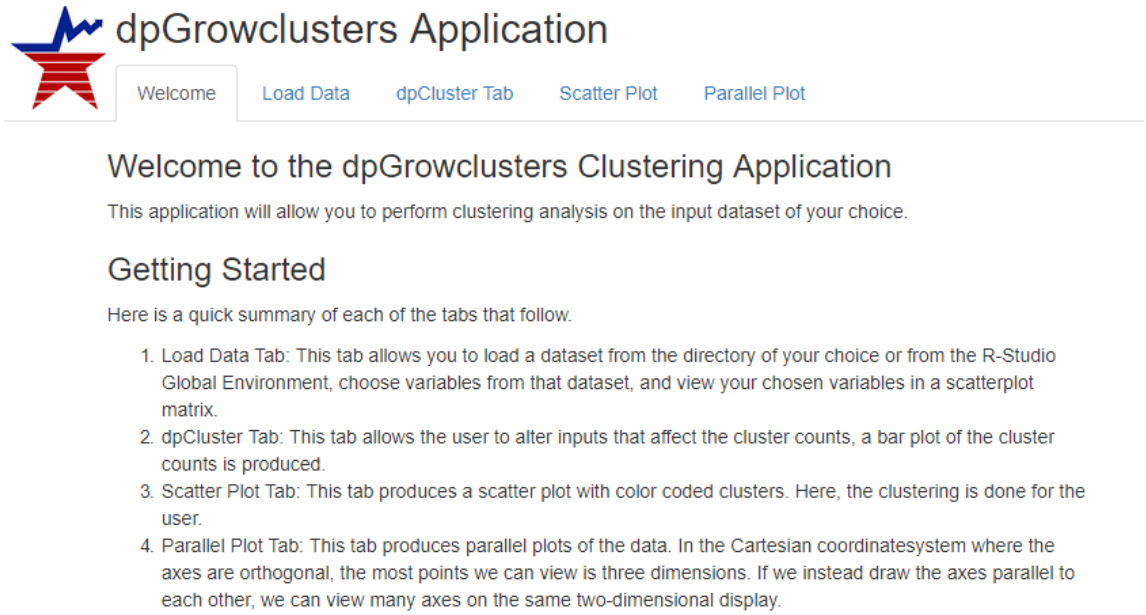

Figure 1. This shows the `dpGrowclusters` application `Welcome` tab. It has information on the various tabs, allowing the user to better understand the workflow and capabilities of the app.

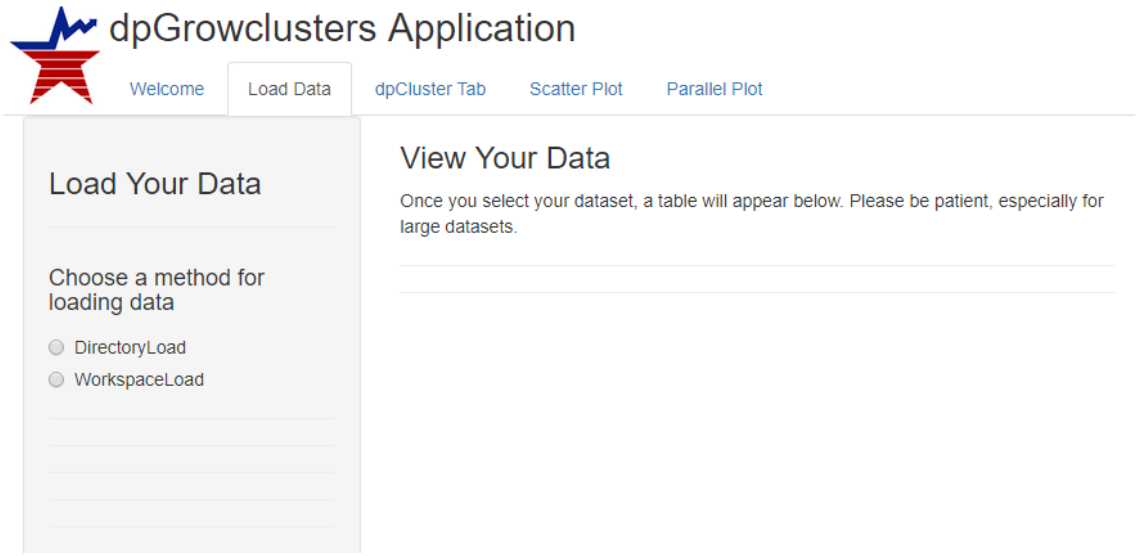

Figure 2. This is a screenshot of the `dpGrowclusters` application `Load Data` tab. Note that the user has the option to load a data set stored in a computer directory or one that is already loaded into the workspace.

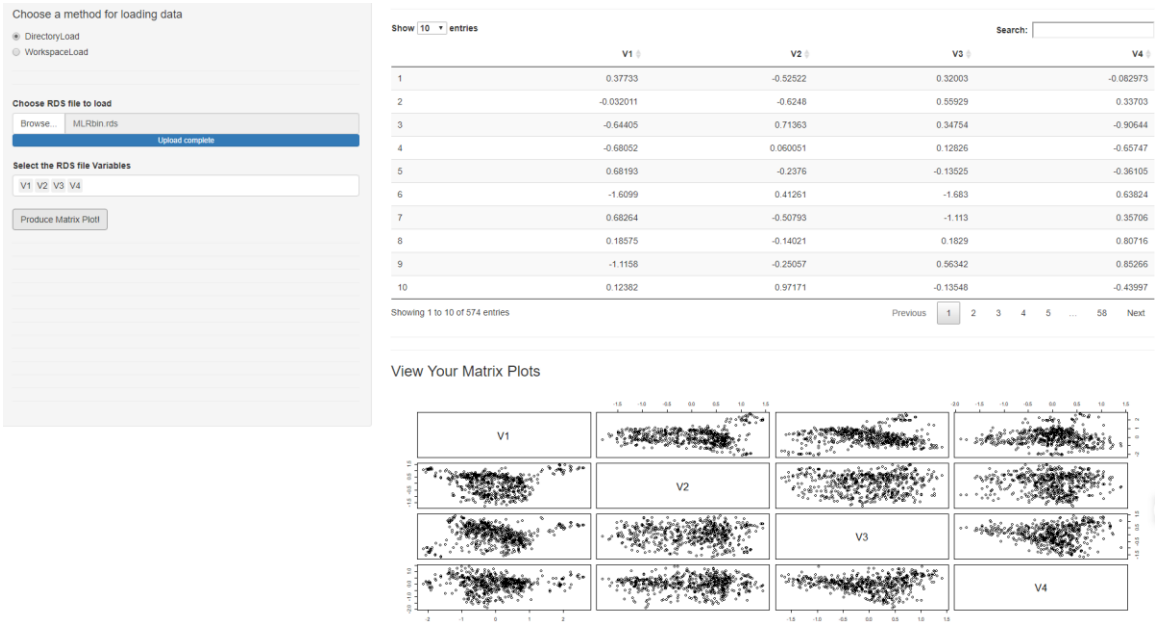

Figure 3. This shows what happens when one loads the MLR (binary) from a directory. Note that the data are shown in a table and in a scatter plot matrix, based on the chosen variables to display (see panel on the left).

Figure 4. This is a screenshot after the user loads the MLR (raw) data from the workspace.

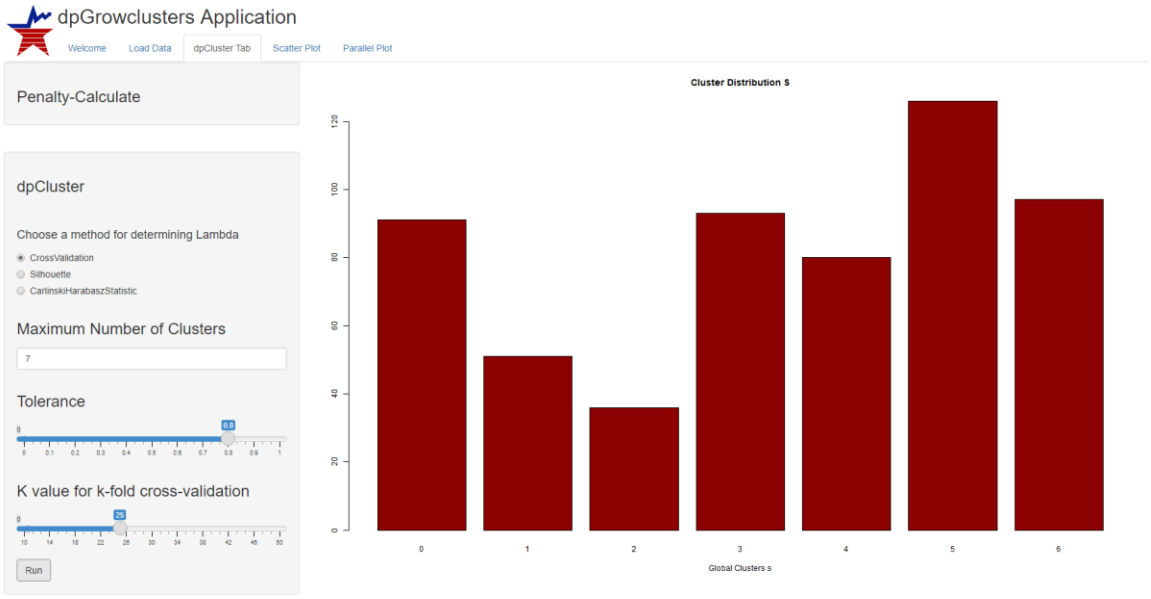

Figure 5. Here is a screenshot of the `dpCluster` tab, showing a bar plot distribution of the clusters using the MLR (raw) data. The user can choose the method for determining the optimal lambda parameter on the left panel. The parameter value impacts the estimated number of clusters.

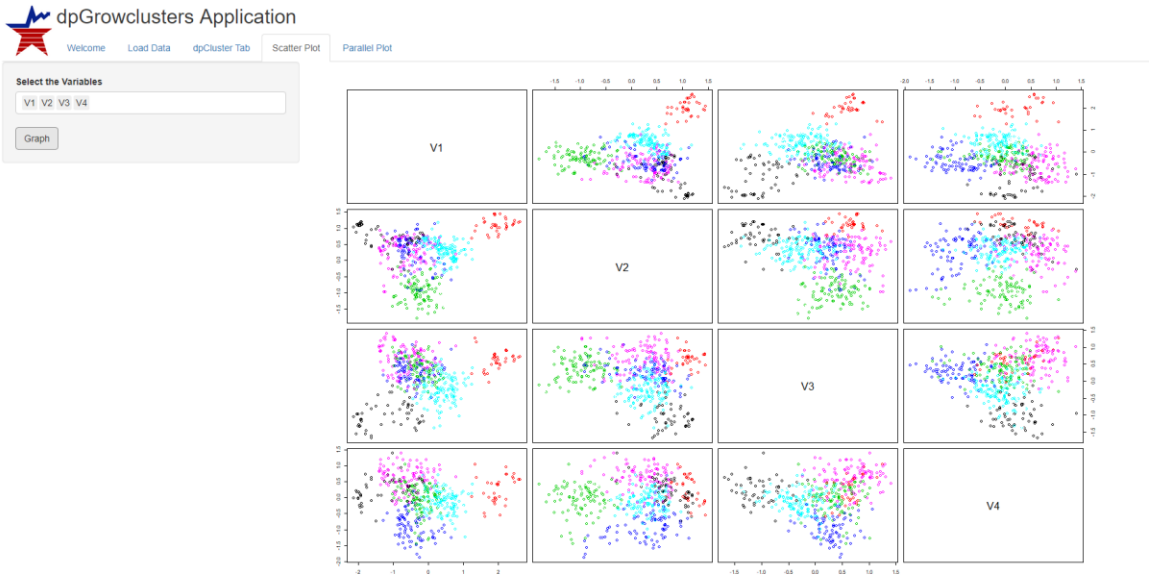

Figure 6. The cluster results can be viewed in the `Scatter Plot` tab. Here we see clusters found in the MLR (raw) data. The colors indicate the cluster, and the groups seem visually reasonable.

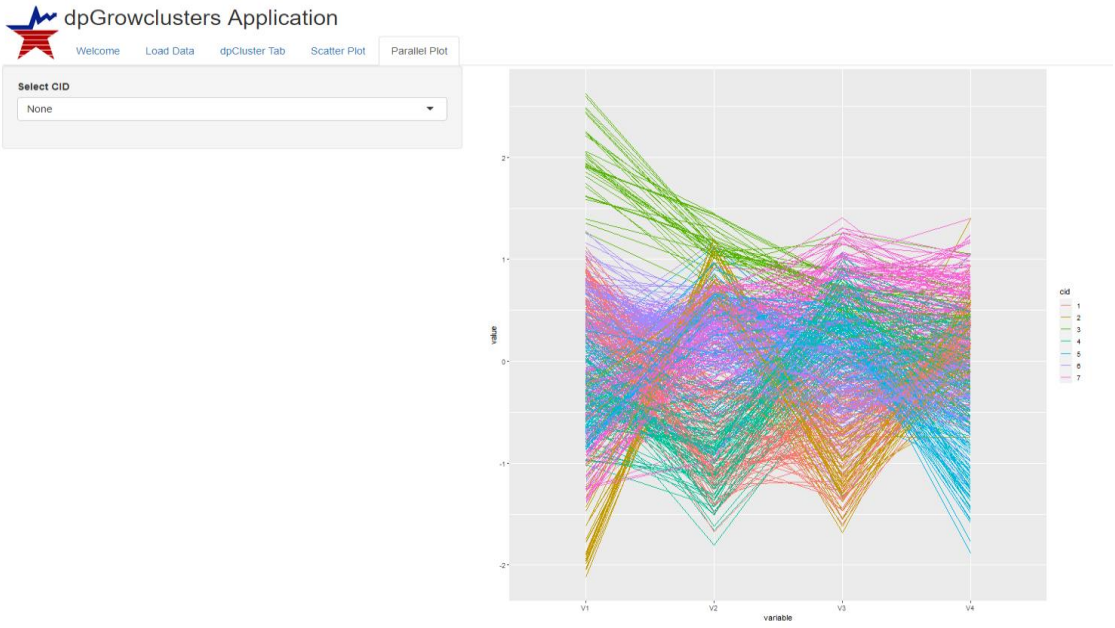

Figure 7. The clusters can be viewed in parallel coordinates at the `Parallel Plot` Tab. This shows the clusters found in the MLR (raw) data, where a color indicates the cluster.

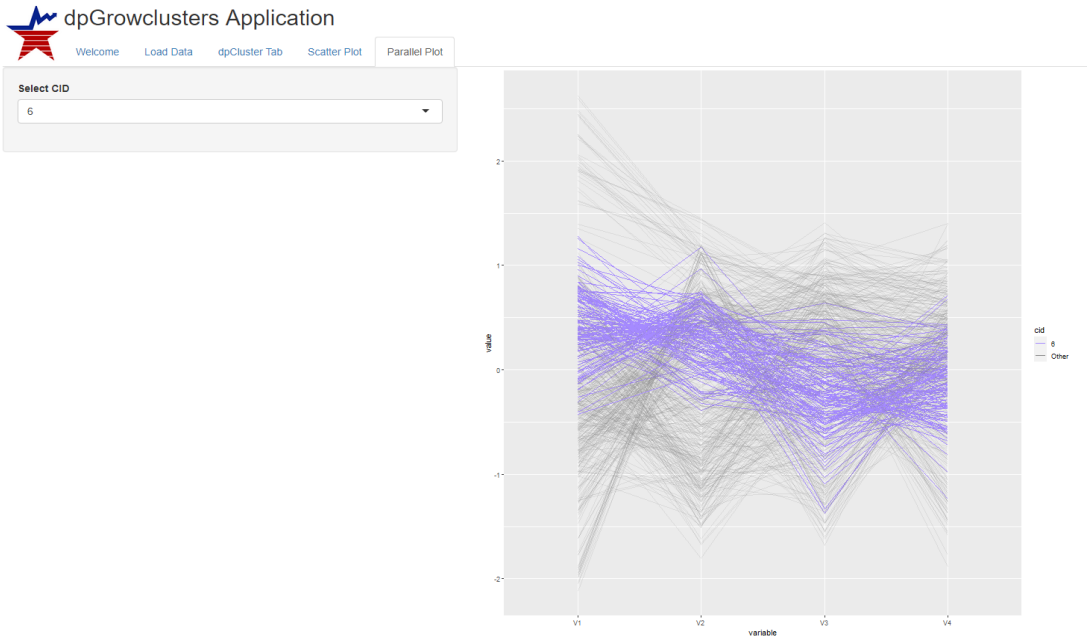

Figure 8. The user can select a cluster to visualize in the dropdown box at the left. This allows one to see if the broken lines in a cluster follow a similar path through the axes and/or are bundled together, indicating observations in the cluster are similar to each other.