Solar Energy 153 (2017) 317–328

Contents lists available at ScienceDirect

Solar Energy

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

Analyzing big time series data in solar engineering using features and PCA

Dazhi Yang [a,*], Zibo Dong [b], Li Hong I. Lim [c], Licheng Liu [d]

[a] Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research (A*STAR), Singapore
[b] Solar Energy Research Institute of Singapore, National University of Singapore, Singapore
[c] Department of Electronic Systems, University of Glasgow, UK
[d] Saferay Pte. Ltd., Singapore

ARTICLE INFO

Article history:
Received 24 April 2017
Received in revised form 21 May 2017
Accepted 24 May 2017

Keywords:
Principal component analysis
Time series features
Solar irradiance
Characterization

ABSTRACT

In solar engineering, we encounter big time series data such as the satellite-derived irradiance data and string-level measurements from a utility-scale photovoltaic (PV) system. While storing and hosting big data are certainly possible using today's data storage technology, it is challenging to effectively and efficiently visualize and analyze the data. We consider a data analytics algorithm to mitigate some of these challenges in this work. The algorithm computes a set of generic and/or application-specific features to characterize the time series, and subsequently uses principal component analysis to project these features onto a two-dimensional space. As each time series can be represented by features, it can be treated as a single data point in the feature space, allowing many operations to become more amenable. Three applications are discussed within the overall framework, namely (1) the PV system type identification, (2) monitoring network design, and (3) anomalous string detection. The proposed framework can be easily translated to many other solar engineer applications.

1. Introduction

Many solar engineering datasets, such as high-resolution satellite-derived irradiance data (e.g., Nikitidou et al., 2015), power output data from hundreds of photovoltaic (PV) plants in an area (e.g., Yang et al., 2017) and module-level measurements from a PV plant (e.g., Guerriero et al., 2016), align well with the HACE theorem[1] proposed by Wu et al. (2014) that characterizes big data. One of the main challenges of processing these raw datasets is the high noise and irrelevant information embedded. Moreover, visualization and analysis through operating directly on the raw datasets can be ineffective. On this point, the Pareto principle, better-known as the 80/20 rule, commonly applies: researchers and solar engineers often spend most of their time collecting, cleaning, filtering, reducing and formatting the data. In this paper, a data analytics algorithm is used to overcome some of the aforementioned challenges. We will look into a class of applications which involve big time series data, or more specifically, solar irradiance and other related forms.

A time series is a collection of observations taken sequentially in time; this definition provides a natural grouping for the data. Instead of viewing the data points as individual entities, we can view time series as *entities*. Once this seemingly trivial statement is understood, much convenience can be added to data handling and analytics. Traditionally, to reduce the complexity in time series data, we often shorten each time series, but preserve the number of entities. For example, satellite-derived irradiance data can be considered as time series of lattice processes. As the data usually span decades, some reduced form, such as a typical meteorological year (TMY) file, can be useful. Composition of the TMY data typify conditions at a particular site over a longer period of time, i.e., 10–30 years. For computer simulations of solar energy conversion systems and building systems to facilitate performance comparisons of different designs, this type of reduced dataset is sufficient (Wilcox and Marion, 2008). Our approach of representing raw time series is similar to the construction of TMY datasets.

The core concept is rather simple: each time series is treated as an individual entity which can be characterized by a set of generic or application-specific features. This step dramatically reduces the dimension of the data, i.e., from hundreds of samples in a time series to a few descriptive features. As each time series can now be

* Corresponding author.
E-mail addresses: yangdazhi.nus@gmail.com, yangdz@simtech.a-star.edu.sg (D. Yang).

[1] Big Data starts with large-volume, **h**eterogeneous, **a**utonomous sources with distributed and decentralized control, and seeks to explore **c**omplex and **e**volving relationships among data (Wu et al., 2014).

http://dx.doi.org/10.1016/j.solener.2017.05.072
0038-092X/

treated as a single data point in the feature space, many operations become more amenable in that feature space. Furthermore, it is also easier to visualize big time series data in the feature space as compared to the traditional time series visualization methods such as the spaghetti plot and horizon plot, which are informative but not very scalable. We illustrate these points with a toy example.

1.1. A toy example

Let us consider the problem of detecting faulty strings using commonly available data from a PV plant. Suppose we represent each string-level output current time series with a single feature, namely the mean value over a period of time, and plot it on the real line, the faulty string could be detected by locating the outliers in that one-dimensional feature space, as illustrated in Fig. 1. While the single feature approach may allow us to detect the faulty strings, it is difficult to isolate the fault type. The decrease in output current is a shared observation for several different fault types (see Table 2 in Chine et al., 2016). If a second feature is added, namely, the mean output voltage over that period of time, the faulty strings can now be represented in a two-dimensional space. Since the voltage of the faulty strings can increase, decrease or remain constant, corresponding to different faults, combining two features would better identify fault types.

The above idea can immediately be expanded to a feature space with p dimensions, with the additional features being, e.g., mean short circuit current, mean open circuit voltage and number of maximum power points; these features were used in Chine et al. (2016). Within the narrow premise of this toy example, more features imply better isolation of faults. In the ideal case, the described approach could circumvent the tedious string-by-string fault check, which often involves complex procedure and flow chart.

The one-dimensional case displayed in Fig. 1 provides excellent visualization of multiple time series. It is not difficult to imagine such plots in a two- or three-dimensional space. When $p > 3$, the scatter plot can still be visualized by performing principal component analysis (PCA) on the features. PCA uses an orthogonal transformation to convert possibly correlated features to linearly uncorrelated variables, known as principal components (PCs). As the first few PCs often contain most variation, it is common to plot the data points – recall each data point represents a time series in our case – in a new low-dimensional space constructed by the directions of the first few PCs. When PCA is considered, its companion algorithms, such as the k-means clustering, α-hull and high density region, can be then applied to solve a variety of problems, and thus make the data analytics algorithm very versatile.

1.2. Applications

We study three applications in this work, namely, (1) PV system type identification, (2) monitoring network design and (3) anomalous PV string detection. We note that all three applications are well-studied in the literature (the literature review will distribute to respective sections), however, the merit of the present work goes to the new point of view on data handling. In clustering problem like the first two applications, the k-means algorithm will be used together with PCA. Unlike other alternatives, this approach does not cluster raw point values using a distance metric, rather it clusters based on global features extracted from each time series. The third application is in line with the toy example above. A two-dimensional outlier detection algorithm, the α-hull algorithm, will be applied to the result of PCA. This is also distinct from most outlier detection studies in the literature, where outliers are identified within one time series or based on statistical rules. Besides the these applications, there are many other applications that could potentially benefit from the analytics algorithm. We briefly enumerate several other applications in Appendix E.

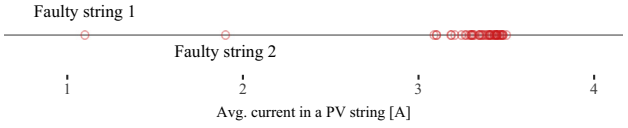

Fig. 1. An illustrative example of PV string fault detection in a one-dimensional feature space.

2. Principal component analysis and biplot

For a *centered* dataset $\boldsymbol{X}$, an $n \times p$ matrix, where n is the number of time series (observation, each time series is considered as one observation) and p is the number of time series features (variable), PCA computes the most meaningful[2] *basis* to re-express $\boldsymbol{X}$. If $\boldsymbol{Z}$ is the re-represented data, the above statement can be written as $\boldsymbol{Z} = \boldsymbol{XA}$, where $\boldsymbol{A}$ is an $p \times p$ matrix and its columns are a set of basis vectors for representing of columns of $\boldsymbol{X}$.

PCA assumes all basis vectors are orthonormal. It first selects a normalized direction in p-dimensional space along which the variance in $\boldsymbol{X}$ is maximized; this basis vector is denoted as $\boldsymbol{a}_1$. In other words, we maximize $\mathbb{V}(\boldsymbol{a}_1^\top \boldsymbol{x})$, where $\boldsymbol{x}$ is vector of p random variables (p time series features in this case). Since the maximum will not be achieved with finite $\boldsymbol{a}_1$, a normalization constraint is imposed, namely, $\boldsymbol{a}_1^\top \boldsymbol{a}_1 = 1$. The subsequent direction is again selected based on the maximum variance criterion, however, due to the orthonormal assumption, the choice is limited to the directions that are perpendicular to $\boldsymbol{a}_1$. The procedure continues until p directions are selected. Thus $\boldsymbol{a}_k^\top \boldsymbol{x}$ is defined as the kth sample principal components and $z_{ik} = \boldsymbol{a}_k^\top \boldsymbol{x}_i$ is the score for the ith observation on the kth PC.

2.1. Solving PCA with eigendecomposition

As the goal of PCA is to reduce redundancy, it is desired that each variable co-varies as little as possible with other variables. In other words, we aim to diagnolize the covariance matrix of the re-represented data. Let $\boldsymbol{S_Z}$ be the covariance matrix of $\boldsymbol{Z}$, i.e.,

$$\boldsymbol{S_Z} = \frac{1}{n-1}\boldsymbol{Z}^\top \boldsymbol{Z}, \tag{1}$$

we have

$$\begin{aligned} \boldsymbol{S_Z} &= \tfrac{1}{n-1}(\boldsymbol{XA})^\top (\boldsymbol{XA}) \\ &= \tfrac{1}{n-1}\boldsymbol{A}^\top (\boldsymbol{X}^\top \boldsymbol{X})\boldsymbol{A} \\ &= \tfrac{1}{n-1}\boldsymbol{A}^\top (\boldsymbol{EDE}^{-1})\boldsymbol{A}, \end{aligned} \tag{2}$$

where $\boldsymbol{E}$ is a matrix of eigenvectors of $\boldsymbol{X}^\top \boldsymbol{X}$ arranged as columns and $\boldsymbol{D}$ is a diagonal matrix. If we let $\boldsymbol{A} \equiv \boldsymbol{E}$, the covariance matrix

$$\begin{aligned} \boldsymbol{S_Z} &= \tfrac{1}{n-1}\boldsymbol{A}^\top (\boldsymbol{ADA}^{-1})\boldsymbol{A} \\ &= \tfrac{1}{n-1}(\boldsymbol{A}^{-1}\boldsymbol{A})\boldsymbol{D}(\boldsymbol{A}^{-1}\boldsymbol{A}) \\ &= \tfrac{1}{n-1}\boldsymbol{D} \end{aligned} \tag{3}$$

can be diagnolized (note that $\boldsymbol{A}^{-1} = \boldsymbol{A}^\top$ when $\boldsymbol{A}$ is orthogonal). This was the goal for PCA. The eigenvectors can be found via eigendecomposition. Alternatively, a more mathematically involved approach to solve PCA is through singular value decomposition

[2] For a detailed discussion on the motivation for PCA, and what should be considered as "most meaningful", we refer the readers to Shlens (2003).