Incremental calculation for multivariate statistics of compositional data

Huiwen Wang
School of Economics and Management, Beihang University, Beijing 100191, China - wanghw@vip.sina.com

Yuan Wei*
School of Economics and Management, Beihang University, Beijing 100191, China - selindaqq@gmail.com

Abstract

Compositional data, containing structure information, have a wide application in various fields. Many statistical models have been extended to compositional data in existed works. However, so far no research has been done for the incremental calculation of these models. With data collection becoming faster and easier through networks, incremental calculation algorithm for compositional data is necessary. In this paper, we give the online updating methods for basic operations of compositional-data vectors, including addition, scalar multiplication and inner production. Based on this, the incremental calculation for some frequently used sample statistics are proposed. Then, plenty of statistical methods that are based on these sample statistics can be calculated incrementally. We take ordinary least squares regression and principle component analysis as examples to design the online updating algorithms for compositional data. The simulation results demonstrate the efficiency of the incremental computing algorithms.
Keywords: Online updating; Simplex space; OLS regression; PCA.

1. Introduction

Compositional data are defined as quantitative description of structure information, which makes it have a wide range of applications in the field of socio-economic management, natural sciences, engineering technology and so forth(Jackson, 1997). For example, in economic analysis, compositional data can be used to reflect the industrial structure or investment components(Pawlowsky-Glahn and Buccianti, 2011). In real life application, with the development of wearable devices, our diet structure, time allocation, categories of activities and other recorded data can all be expressed as compositional data.
All components of compositional data are subject to non-negative and constant-sum constraints(Pawlowsky-Glahn and Buccianti, 2011). This property makes standard statistical methods to be inappropriate to analyse this type of data. The general idea is to release the constrains first by transformation, and then classical statistical methods can be applied on the transformed data. A family of logratio transformations has been introduced (Aitchison, 1982; Filzmoser and Hron, 2008), including the additive logratio (alr) transformation, the centered logratio (clr) transformation, and the isometric logratio (ilr) transformation. Plenty of models are built based on transformation data, such as principle component analysis (PCA)(Aitchison, 1983), ordinary least squares (OLS) regression(Aitchison, 1986), discriminant analysis(Lachenbruch, 1975) and so on. Wang et al. (2013) proposed the inner product for compositional-data vectors, and upon this they showed the completely equivalence of OLS regression on transformed data and compositional data by experiments. Some other works that direct modelling on compositional data for multivariate statistics can be found in Filzmoser et al. (2012);Filzmoser et al. (2009);Filzmoser and Hron (2009).
Recently, with the popularity of the network, data continuously arrives in massive, multiple, rapid and time-varying streams. It is essential for companies to update their models timely for daily works. For instance, the E-commerce websites build their analysis models on the changing structure of user characteristics, which can help them to understand users timely(Russell, 2013). However, as it is obviously not practical to save all these history datasets, incremental calculation (or online updating) becomes indispensable and has captured more and more attention in many applications(Muthukrishnan, 2005). The research of incremental calculation aroused peoples concern since 2000, when Giraud-Carrier (2000) defined the notion of incrementality for learning tasks and algorithms. Up to now, works in this area mainly focus on two directions:

incremental algorithms and incremental computing frameworks. Gaber et al. (2005) categorized the various incremental algorithms to data-based solutions and task-based solutions. Data-based solutions refer to choosing a subset or some transformation of the whole data, such as sampling(Domingos and Hulten, 2001), load shedding(Mayur et al., 2003), sketching(Muthukrishnan, 2005) and others. This kind of methods has the drawback of accuracy, and it is also difficult to determine the right subset and sample size. Task-based solutions are more extensive in research works, such as Mairal et al. (2010); Chu et al. (2007).
The previous incremental algorithms are about real numbers. And to our best knowledge, there have no online updating algorithms for compositional data. In this paper, we propose the incremental calculation for compositional-data vectors, and introduce online updating methods for some sample statistics. Based on these, a plenty of algorithms can achieve incremental computing.
The reminder of the paper is organized as follows. In section 2 first we introduce the operations of compositional data and vectors. Then, the incremental calculation methods for compositional data operations are proposed. Section 3 deduces incremental computing methods for some sample statistics, and upon which we design online updating algorithm for two frequently used statistical methods. This is followed by simulations on two generated data sets in section 4. Section 5 concludes the paper.

2. Incremental calculation for operations

To start with, in this section we present some basic notations for compositional data. And based on this, we propose the incremental calculation for some frequently used operations of compositional-data vectors.

2.1 Preliminaries

In general, a Simplex space of D parts is defined as: $S^D = \{\boldsymbol{x} = (x_1, x_2, \ldots, x_D) \mid x_j > 0,\ j = 1, 2, \ldots, D;\ \sum_{j=1}^{D} x_j = 1\}$ Given two simplex data $\boldsymbol{x}, \boldsymbol{y} \in S^D$, which are referred to as D-part compositional data, Aitchison (1986) stated the linear operation of $\boldsymbol{x}, \boldsymbol{y}$:

$$\boldsymbol{x} \oplus \boldsymbol{y} = \text{Ç}(x_1 y_1, x_2 y_2, \ldots, x_D y_D), \quad \beta \otimes \boldsymbol{x} = \text{Ç}(x_1^{\beta}, x_2^{\beta}, \ldots, x_D^{\beta}),\ \forall \beta \in R. \tag{1}$$

where Ç denotes the closure operator: $\text{Ç}(x_1, x_2, \ldots, x_D) = \left[\frac{x_1}{\sum_{i=1}^{D} x_i}, \frac{x_2}{\sum_{i=1}^{D} x_i}, \ldots, \frac{x_D}{\sum_{i=1}^{D} x_i}\right]$. Then, $\boldsymbol{x} \ominus \boldsymbol{y}$ can be deduced as: $\boldsymbol{x} \ominus \boldsymbol{y} = \boldsymbol{x} \oplus ((-1) \otimes \boldsymbol{y}) = \text{Ç}\left(\frac{x_1}{y_1}, \frac{x_2}{y_2}, \ldots, \frac{x_D}{y_D}\right)$. Mapping $\boldsymbol{x}, \boldsymbol{y}$ to $ilr(\boldsymbol{x}), ilr(\boldsymbol{y})$ with isometric logratio (ilr) transformation(Egozcue et al., 2003), and get the inner product:

$$(\boldsymbol{x}, \boldsymbol{y})_s = < ilr(\boldsymbol{x}), ilr(\boldsymbol{y}) > = \sum_{k=1}^{D-1} \left(\frac{k}{k+1} log \frac{\sqrt[k]{\prod_{j=1}^{k} x_j}}{x_{k+1}} log \frac{\sqrt[k]{\prod_{j=1}^{k} y_j}}{y_{k+1}} \right) \tag{2}$$

Wang et al. (2013) proposed operations for compositional-data vectors by using a component-wise manner. Define $\boldsymbol{u}_j = (\boldsymbol{u}_{1j}, \boldsymbol{u}_{2j}, \ldots, \boldsymbol{u}_{nj})^T,\ j = 1, 2, \ldots, p$ with D-part compositional data $\boldsymbol{u}_{ij} \in S^D$. Then, $\boldsymbol{u}_j \in S^{D_n} = S^D \times S^D \times \ldots \times S^D$. Operations of n-dimensional compositional data vectors can be deduced:

$$\begin{aligned}
\boldsymbol{u}_j \oplus \boldsymbol{u}_k &= (\boldsymbol{u}_{1j} \oplus \boldsymbol{u}_{1k}, \boldsymbol{u}_{2j} \oplus \boldsymbol{u}_{2k}, \ldots, \boldsymbol{u}_{nj} \oplus \boldsymbol{u}_{nk})^T & (3)\\
\beta \otimes \boldsymbol{u}_j &= (\beta \otimes \boldsymbol{u}_{1j}, \beta \otimes \boldsymbol{u}_{2j}, \ldots, \beta \otimes \boldsymbol{u}_{nj})^T,\ \forall \beta \in R. & (4)\\
\boldsymbol{u}_j \ominus \boldsymbol{u}_k &= (\boldsymbol{u}_{1j} \ominus \boldsymbol{u}_{1k}, \boldsymbol{u}_{2j} \ominus \boldsymbol{u}_{2k}, \ldots, \boldsymbol{u}_{nj} \ominus \boldsymbol{u}_{nk})^T & (5)\\
< \boldsymbol{u}_j, \boldsymbol{u}_k >_{S^{D_n}} &= \sum_{i=1}^{n} (\boldsymbol{u}_{ij}, \boldsymbol{u}_{ik})_S & (6)
\end{aligned}$$

2.2 Additivity of addition, scalar multiplication and inner production

Denote $\boldsymbol{u}_j \in S^{D_n}, \boldsymbol{u}_j^* \in S^{D_m},\ j = 1, 2, \ldots, p$, get $\tilde{\boldsymbol{u}}_j = (\boldsymbol{u}_{1j}, \ldots, \boldsymbol{u}_{nj}, \boldsymbol{u}_{1j}^*, \ldots, \boldsymbol{u}_{mj}^*)^T \in S^{D_{n+m}}$. For convenience, in what follows, we use $\tilde{\boldsymbol{u}}_j = \boldsymbol{u}_j \boxplus \boldsymbol{u}_j^* \in S^{D_{n+m}}$ to represent $\tilde{\boldsymbol{u}}_j$, where "$\boxplus$" is the operation that joins vectors by rows without intersection. It is easy to prove that:

$$\begin{aligned}
\tilde{\boldsymbol{u}}_j \oplus \tilde{\boldsymbol{u}}_k &= (\boldsymbol{u}_j \oplus \boldsymbol{u}_k) \boxplus (\boldsymbol{u}_j^* \oplus \boldsymbol{u}_k^*) & (7)\\
\beta \otimes \tilde{\boldsymbol{u}}_j &= (\beta \otimes \boldsymbol{u}_j) \boxplus (\beta \otimes \boldsymbol{u}_j^*),\ \forall \beta \in R & (8)\\
< \tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{u}}_k >_{S^{D(n+m)}} &= < \boldsymbol{u}_j, \boldsymbol{u}_k >_{S^{D_n}} + < \boldsymbol{u}_j^*, \boldsymbol{u}_k^* >_{S^{D_m}} & (9)
\end{aligned}$$

Equation (7) and (8) indicate that linear operations of $\tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{u}}_k$ can be written as stacking the calculation result of $\boldsymbol{u}_j^*, \boldsymbol{u}_k^*$ to the result of $\boldsymbol{u}_j, \boldsymbol{u}_k$. And Equation (9) presents that the inner product for $\tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{u}}_k$ can be calculated by adding the inner product of $\boldsymbol{u}_j, \boldsymbol{u}_k$ and $\boldsymbol{u}_j^*, \boldsymbol{u}_k^*$. Thus, the above calculations of updated vectors $\tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{u}}_k$ can always be computed separately by $\boldsymbol{u}_j, \boldsymbol{u}_k$ and $\boldsymbol{u}_j^*, \boldsymbol{u}_k^*$, which can ensure the incrementally computing in this vector space.

3. Incremental calculation algorithms

Based on the additivity of the operations proposed in section 2, in this part we design the online updating algorithms for some sample statistics of compositional data. Upon this, a plenty of statistical methods can achieve incremental computing. We take OLS regression and PCA as examples to design their online updating algorithms.

3.1 Updating some sample statistics

For n observations of compositional-data variable $\boldsymbol{u}_j \in S^{D_n}$, the sample mean $\boldsymbol{g}_j \in S^D$ is given by Martn-Fernndez et al. (1998):

$$\boldsymbol{g}_j = \text{Ç}(\text{g}(\mathbf{L}_{j1}), \text{g}(\mathbf{L}_{j2}), \ldots, \text{g}(\mathbf{L}_{jD})) \tag{10}$$

where $\mathbf{L}_{jk} = (u_{1jk}, u_{2jk}, \ldots, u_{njk})^T$, $k = 1, \ldots, D$, and $\text{g}(\cdot)$ is the geometric mean. For newly added $\boldsymbol{u}_j^* \in S^{D_m}$, denote $\tilde{\boldsymbol{u}}_j = \boldsymbol{u}_j \boxplus \boldsymbol{u}_j^* \in S^{D_{n+m}}$. By Equation (3) and (4), it is easy to prove that:

$$\tilde{\boldsymbol{g}}_j = \frac{1}{n+m} \otimes (n \otimes \boldsymbol{g}_j \oplus m \otimes \boldsymbol{g}_j^*) \tag{11}$$

where $\tilde{\boldsymbol{g}}_j, \boldsymbol{g}_j^*$ is the sample mean of $\tilde{\boldsymbol{u}}_j, \boldsymbol{u}_j^*$. So the sample mean can be updated only based on the previous $\boldsymbol{g}_j$ and the sample number n, m. Then, by Equation (5), the updated $\tilde{\boldsymbol{u}}_j$ can be centralized to $\tilde{\boldsymbol{o}}_j$, where $\tilde{\boldsymbol{o}}_j = (\boldsymbol{u}_{1j} \ominus \tilde{\boldsymbol{g}}_j, \ldots, \boldsymbol{u}_{nj} \ominus \tilde{\boldsymbol{g}}_j, \boldsymbol{u}_{1j}^* \ominus \tilde{\boldsymbol{g}}_j, \ldots, \boldsymbol{u}_{mj}^* \ominus \tilde{\boldsymbol{g}}_j)^T$.

The inner product of two compositional-data variables $\boldsymbol{u}_j, \boldsymbol{u}_k$ is given by Wang et al. (2013):

$$< \boldsymbol{u}_j, \boldsymbol{u}_k >_{S^{D_n}} = \sum_{i=1}^{n} (\boldsymbol{u}_{ij}, \boldsymbol{u}_{ik})_S \tag{12}$$

Denote $\tilde{\boldsymbol{o}}_k$ to be the centralized $\tilde{\boldsymbol{u}}_k$, and $\tilde{\boldsymbol{g}}_k$ to be the sample mean of $\tilde{\boldsymbol{u}}_k$. By Equation (7),(9) and (12), the inner product of $\tilde{\boldsymbol{o}}_j, \tilde{\boldsymbol{o}}_k$ can be deduced:

$$\begin{aligned} < \tilde{\boldsymbol{o}}_j, \tilde{\boldsymbol{o}}_k >_{S^{D_{n+m}}} &= \sum_{i=1}^{n+m} (\tilde{\boldsymbol{u}}_{ij} \ominus \tilde{\boldsymbol{g}}_j, \tilde{\boldsymbol{u}}_{ik} \ominus \tilde{\boldsymbol{g}}_k)_S \\ &= \sum_{i=1}^{n} (\boldsymbol{u}_{ij} \ominus \tilde{\boldsymbol{g}}_j,\ \boldsymbol{u}_{ik} \ominus \tilde{\boldsymbol{g}}_k)_S + \sum_{i=1}^{m} (\boldsymbol{u}_{ij}^* \ominus \tilde{\boldsymbol{g}}_j,\ \boldsymbol{u}_{ik}^* \ominus \tilde{\boldsymbol{g}}_k)_S \end{aligned} \tag{13}$$

Define $ilr(\boldsymbol{u}_j) = (ilr(\boldsymbol{u}_{1j}), \ldots, ilr(\boldsymbol{u}_{nj}))^T$(Wang et al., 2013). Equation (13) can be deduced as:

$$< \tilde{\boldsymbol{o}}_j, \tilde{\boldsymbol{o}}_k >_{S^{D_{n+m}}} = < ilr(\boldsymbol{u}_j), ilr(\boldsymbol{u}_k) > + < ilr(\boldsymbol{u}_j^*), ilr(\boldsymbol{u}_k^*) > -(n+m) < ilr(\tilde{\boldsymbol{g}}_j), ilr(\tilde{\boldsymbol{g}}_k) > \tag{14}$$

So the centralized inner product of $\tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{u}}_k$ can be computed by the previous inner product, previous sample mean and the newly added data.

3.2 Updating some statistical algorithms

Based on the incremental computing methods proposed above, we design incremental calculation for two classical statistical algorithms: OLS regression and principle component analysis.

OLS Regression

Denote compositional data matrix $\mathbf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p]$, where $\boldsymbol{x}_j \in S^{D_n}$, $j = 1, 2, \ldots, p$ are independent

variables, and $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n) \in S^{D_n}$ is the dependent variable. Using $\boldsymbol{u}_k, \boldsymbol{v}$ to be the centralized $\boldsymbol{x}_k, \boldsymbol{y}$, the estimator of $\boldsymbol{\beta}$ can be solved from the following equation in a matrix form(Wang et al., 2013):

$$\begin{pmatrix} < \boldsymbol{u}_1, \boldsymbol{u}_1 >_{S^{D_n}} & \ldots & < \boldsymbol{u}_1, \boldsymbol{u}_p >_{S^{D_n}} \\ \vdots & \ddots & \vdots \\ < \boldsymbol{u}_p, \boldsymbol{u}_1 >_{S^{D_n}} & \ldots & < \boldsymbol{u}_p, \boldsymbol{u}_p >_{S^{D_n}} \end{pmatrix} \begin{pmatrix} \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_p \end{pmatrix} = \begin{pmatrix} < \boldsymbol{u}_1, \boldsymbol{v} >_{S^{D_n}} \\ \vdots \\ < \boldsymbol{u}_p, \boldsymbol{v} >_{S^{D_n}} \end{pmatrix} \tag{15}$$

For new coming samples $\mathbf{X}^*, \boldsymbol{y}^*$, write the updated dataset as $\tilde{\mathbf{X}} = \mathbf{X} \boxplus \mathbf{X}^*$, $\tilde{\boldsymbol{y}} = \boldsymbol{y} \boxplus \boldsymbol{y}^*$. We use $\boldsymbol{g}_{(x)j}, \boldsymbol{g}_{(y)}$ to represent the sample mean of $\boldsymbol{x}_j, \boldsymbol{y}$. Denote $j = 1, \ldots, p;\ k = 1, \ldots, p$. Then, we can achieve incremental calculation by Algorithm 1.

Algorithm 1: Incremental calculation for OLS Regression

Initialize $\boldsymbol{g}_{(x)j}$, $\boldsymbol{g}_{(y)}, < \boldsymbol{x}_j, \boldsymbol{x}_k >_{S^{D_n}}$, $< \boldsymbol{x}_j, \boldsymbol{y} >_{S^{D_n}}, n$ from previous computing results.
Read in new $\mathbf{X}^*, \boldsymbol{y}^*$:
(1) Calculate $\boldsymbol{g}^*_{(x)j}, \boldsymbol{g}^*_{(y)}$ by Equation (10).
(2) Calculate $< \tilde{\boldsymbol{x}}^*_j, \tilde{\boldsymbol{x}}^*_k >_{S^{D_m}}, < \tilde{\boldsymbol{x}}^*_j, \tilde{\boldsymbol{y}}^* >_{S^{D_m}}$ by Equation (12).
(3) Update sample mean $\boldsymbol{g}_{(x)j}, \boldsymbol{g}_{(y)}$ to $\tilde{\boldsymbol{g}}_{(x)j}, \tilde{\boldsymbol{g}}_{(y)}$ using Equation (11).
(4) Update the elements in (15) to $< \tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{u}}_k >_{S^{D_{n+m}}}, < \tilde{\boldsymbol{u}}_j, \tilde{\boldsymbol{v}} >_{S^{D_{n+m}}}$ by Equation (14).
(5) Solve $\hat{\boldsymbol{\beta}}$ by the equation in the updated matrix form (15).

Principle Component Analysis

Wang et al. (2013) proved that the derivation of PCA for multiple compositional data variables could be transformed into the eigen-decomposition of the covariance matrix $\mathbf{W}$, when the variables are similar in terms of range and scale or in the same units of measure. In this situation, we can update $\mathbf{W}$ incrementally by the method including in Algorithm 1, and then eigenvalues and eigenvectors can be computed from matrix decomposition :

$$\tilde{\mathbf{W}} \tilde{\boldsymbol{\mu}} = \tilde{\boldsymbol{\lambda}} \tilde{\boldsymbol{\mu}} \tag{16}$$

With the updated $\tilde{\boldsymbol{\lambda}}$, $\tilde{\boldsymbol{\mu}}$, the first K components for new coming data $\mathbf{X}^*$ can be calculated as:

$$\mathbf{F}^*_k = \oplus_{j=1}^{p} (\tilde{\boldsymbol{\mu}}_{kj} \otimes \mathbf{x}^*_j) \tag{17}$$

The theoretical time complexity analysis for the two non-incremental algorithms is $O(np^2 + p^3)$(Chu et al., 2007). With incremental calculation, the time complexity is $O(n^* p^2) + O(p^3)$, where n^* is the new coming sample size. When $n \gg p$, which is common in real applications, the incremental calculation will make a big difference.

4. Simulation studies

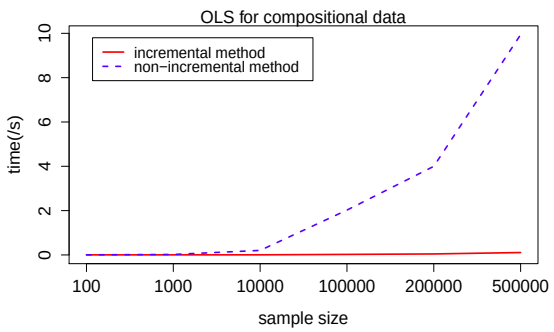
The purpose of this section is to show the efficiency of the incremental calculation for compositional data. To provide clear comparison, each algorithm had two different versions: One using the incremental calculation, and the other using non-incremental method. For incremental calculation, we use algorithms designed in section 3, and for the non-incremental method use the algorithms in Wang et al. (2013, 2015). We take OLS regression and PCA as examples to show the advantages of incremental calculation.

4.1 Simulation 1

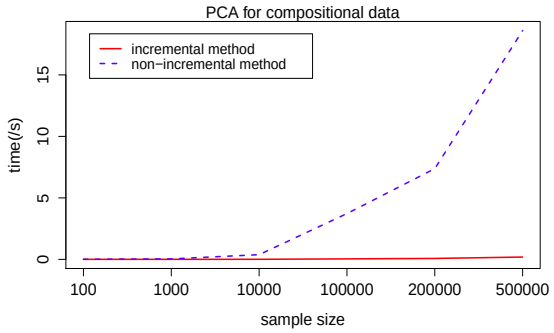
In this part, we conduct the simulation of OLS regression method. Create $\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \epsilon_1, \epsilon_2 \sim N(0, 1)$, $(i = 1, 2, 3)$. By *ilr*-transformation, 3-part compositional data is generated, which is $\boldsymbol{x}_i = (\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \boldsymbol{x}_{i3})$, $\epsilon = (\epsilon_1, \epsilon_2, \epsilon_3) \in S^3$, $(i = 1, 2, 3)$. And get $\boldsymbol{y} = 0.49 \otimes \boldsymbol{x}_1 \oplus 0.26 \otimes \boldsymbol{x}_2 \oplus 1.6 \otimes \boldsymbol{x}_3 \oplus \epsilon$.

In order to compare the improved efficiency under different dataset size, we use random sampling method to draw $(100, 1000, 10000, 100000, 200000, 500000)$ samples respectively in each experiment and repeat 30 times under each sample size. We set the increasing sample number as 1 percent of the original sample

size. The environment that the experiments conducted on is an Intel i5 3.20GHz CPU and 3556MB physical memory. The operating system is Windows XP. Using Python programming language environment, the average calculation time are recorded in Fig.1(a).



(a) Computing time for OLS regression



(b) Computing time for PCA

The X-axis of Fig.1(a) represents different sample size range from 100 to 500000, and Y-axis represents the time used in computing. Two lines are drawn in the graph for incremental calculation and non-incremental calculation respectively. The modeling time of two calculation methods begins to be different when sample size comes up to 10000. And as the incremental calculation is not affected by the original sample size, the advantages will be further expanded with increased size.
We compare the estimated $\hat{\beta}$ of the two calculation methods. Take the simulation of 10000 sample size as example, the average difference between the two methods for $\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3$ under 30 replications are $0.0005, 0.0002, -0.0001$ respectively, and the difference standards are $0.0009, 0.0009, 0.0007$. As the incremental calculation actually is theoretically equivalent to non-incremental calculation, this result is reasonable.

4.2 Simulation 2
In this simulation, we conduct the simulation for PCA method. First create two independent hidden factors $V_1 \sim N(0, 290),\ V_2 \sim N(0, 300)$. Then $x_{i1}, x_{i2},\ (i = 1, 2, \ldots, 6)$ are generated as: (1) $\boldsymbol{x}_{i1} = V_1 + \epsilon_i^1, \boldsymbol{x}_{i2} = V_1 + \epsilon_i^2,\ \epsilon_i^1, \epsilon_i^2 \sim N(0, 1)\ i = 1, 2, 3$; (2) $\boldsymbol{x}_{i1} = V_2 + \epsilon_i^1,\ \boldsymbol{x}_{i2} = V_2 + \epsilon_i^2,\ \epsilon_i^1, \epsilon_i^2 \sim N(0, 1),\ i = 4, 5, 6$
Then generate 3-part compositional data by *inverse-ilr* transformation. We use the same experiment design and environment with simulation 1. The calculation time is recorded in Fig.1(b). Similar to OLS regression, the modeling time for two calculation methods demonstrates the efficiency of incremental computing. In the experiment of 10000 sample size, the experienced variance of the first two eigenvalues is bigger than 99 percent for both calculation method. We give the first two eigenvectors in Tab.1, which proves the effectiveness of the incremental methods.

Table 1: The first two eigenvectors for incremental and non-incremental calculation

		$\boldsymbol{x}_1$	$\boldsymbol{x}_2$	$\boldsymbol{x}_3$	$\boldsymbol{x}_4$	$\boldsymbol{x}_5$	$\boldsymbol{x}_6$
Eigenvector 1	incremental	-0.5633	-0.5635	-0.5630	0.1267	0.1268	0.1267
	non-incremental	-0.5667	-0.5670	-0.5664	0.1104	0.1106	0.1103
Eigenvector 2	incremental	-0.1263	-0.1269	-0.1270	-0.5625	-0.5636	-0.5637
	non-incremental	-0.1100	-0.1106	-0.1107	-0.5660	-0.5670	-0.5671

5. Conclusions

In this paper, we first propose the incremental calculation for compositional data, which is of importance at present. The additivity of some basic operations is presented, and the incremental calculation for some frequently used sample statistics is deduced. Based on this, many statistical methods can achieve incremental computing. We take OLS regression and PCA as examples to design the online updating algorithms. The simulation results show that incremental computing can save both memory space and calculation time.

6. Acknowledgements

This work was partially supported by the National Natural Science Foundation of China(Grant No. 7142010725) and the National High Technology Research and Development Program of China(SS2014AA012303).

Reference

Aitchison, J. (1982). The statistical analysis of compositional data. *Journal of the Royal Statistical Society. Series B (Methodological)*, 139–177.

Aitchison, J. (1983). Principal component analysis of compositional data. *Biometrika 70*(1), 57–65.

Aitchison, J. (1986). The statistical analysis of compositional data.

Chu, C., S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun (2007). Map-reduce for machine learning on multicore. *Advances in neural information processing systems 19*, 281.

Domingos, P. and G. Hulten (2001). A general method for scaling up machine learning algorithms and its application to clustering. pp. 106–113.

Egozcue, J. J. E., V. Pawlowsky-Glahn, G. O. R. Mateu-Figueras, and C. Barcel O Vidal (2003). Isometric logratio transformations for compositional data analysis. *Mathematical Geology 35*(3), 279–300.

Filzmoser, P. and K. Hron (2008). Outlier detection for compositional data using robust methods. *Mathematical Geosciences 40*(3), 233–248. 44.

Filzmoser, P. and K. Hron (2009). Correlation analysis for compositional data. *Mathematical Geosciences 41*(8), 905–919.

Filzmoser, P., K. Hron, and C. Reimann (2009). Univariate statistical analysis of environmental (compositional) data: Problems and possibilities. *Science of The Total Environment 407*(23), 6100 – 6108.

Filzmoser, P., K. Hron, and M. Templ (2012). Discriminant analysis for compositional data and robust parameter estimation. *Computational Statistics 27*(4), 585–604.

Gaber, M. M., A. Zaslavsky, and S. Krishnaswamy (2005). Mining data streams: a review. *ACM Sigmod Record 34*(2), 18–26.

Giraud-Carrier, C. (2000). A note on the utility of incremental learning. *AI Communications 13*(4), 215–223.

Jackson, D. A. (1997). Compositional data in community ecology: The paradigm or peril of proportions? *Ecology 78*(3), 929–940.

Lachenbruch, P. A. (1975). *Discriminant analysis*. Wiley Online Library.

Mairal, J., F. Bach, J. Ponce, and G. Sapiro (2010). Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res. 11*, 19–60.

Martn-Fernndez, J. A., C. Barcel-Vidal, and V. Pawlowsky-Glahn (1998). A critical approach to non-parametric classification of compositional data. In A. Rizzi, M. Vichi, and H.-H. Bock (Eds.), *Advances in Data Science and Classification*, Advances in Data Science and Classification, pp. 49–56. Springer Berlin Heidelberg.

Mayur, B. B., B. Babcock, M. Datar, and R. Motwani (2003). Load shedding techniques for data stream systems.

Muthukrishnan, S. (2005). *Data streams: Algorithms and applications*. Now Publishers Inc.

Pawlowsky-Glahn, V. and A. Buccianti (2011). *Compositional data analysis: Theory and applications*. John Wiley & Sons.

Russell, M. A. (2013). *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More*. " O'Reilly Media, Inc.".

Wang, H., L. Shangguan, R. Guan, and L. Billard (2015). Principal component analysis for compositional data vectors. *Computational Statistics* (In press).

Wang, H., L. Shangguan, J. Wu, and R. Guan (2013). Multiple linear regression modeling for compositional data. *Neurocomputing 122*, 490–500.