Sparse Multiple Factor Analysis, sparse STATIS, and sparse DiSTATIS with applications to sensory evaluation

Ju-chi Yu, Carlos Gómez-Corona, Hervé Abdi, Vincent Guillemot

Received: 6 May 2022 | Accepted: 13 September 2022

DOI: 10.1002/cem.3443

SPECIAL ISSUE - RESEARCH ARTICLE

Journal of CHEMOMETRICS WILEY

Sparse Multiple Factor Analysis, sparse STATIS, and sparse DiSTATIS with applications to sensory evaluation

Ju-Chi Yu[1] | **Carlos Gómez-Corona**[2] | **Hervé Abdi**[3] | **Vincent Guillemot**[4]

[1]Campbell Family Mental Health Research Institute, Centre for Addiction and Mental Health, Toronto, Canada

[2]Firmenich & Cie, Neuilly-sur-Seine, France

[3]The University of Texas at Dallas, Richardson, Texas, USA

[4]Institut Pasteur, Université Paris Cité, Bioinformatics and Biostatistics Hub, Paris, France

Correspondence

Ju-Chi Yu, Campbell Family Mental Health Research Institute, Centre for Addiction and Mental Health, Toronto, Canada.
Email: Ju-Chi.Yu@camh.ca

Hervé Abdi, The University of Texas at Dallas, Richardson, TX, USA.
Email: herve@utdallas.edu

Vincent Guillemot, Institut Pasteur, Université Paris Cité, Bioinformatics and Biostatistics Hub, F-75015 Paris, France.
Email: vincent.guillemot@pasteur.fr

Abstract

Component-based multitable methods, such as multiple factor analysis (MFA), STATIS, and DiSTATIS, are routinely used to analyze multiblock data, which are now common in chemometrics and sensory evaluation studies. These blocks of data form data tables that—for example, in sensory evaluation—describe how different assessors evaluate a set of products either on a set of descriptors or on the similarity between products. To analyze these data, component-based multitable methods extract orthogonal components explaining most of the variance of the data. However, when the data tables are heterogeneous or have complex structures, a single component space does not represent the data well and can give components that are difficult to interpret. Previous literature solved this interpretation problem by eliminating irrelevant variables—a process called *sparsification*—while keeping the components orthogonal. Here, we extended such methods to develop sparsification algorithms for three multitable methods, namely, "sparse MFA" (sMFA), "sparse STATIS" (sSTATIS), and "sparse DiSTATIS" (sDiSTATIS). In these sparse methods, we sparsified the data tables to identify the most informative assessors or products. In sMFA, we show how group sparsity can be used to sparsify whole tables (i.e., assessors or products), hereby greatly increasing the interpretability of sMFA's outcome. In sSTATIS and sDiSTATIS, we developed two different sparsification approaches: One approach creates subgroups of products and simplifies the components to facilitate interpretation; whereas the other approach creates subgroups of assessors and alleviates the problem of heterogeneity. We showed with three examples how these sparse methods increase interpretability of the results in sensory evaluation.

KEYWORDS

multiblock, sensory evaluation, sparsification

1 | INTRODUCTION

In chemometrics and sensory evaluation, studies often involve data—called multitable data—comprising several subtables each corresponding to a specific group of variables. For example, in sensory evaluation, each data table could describe how one participant evaluated a set of products. Two among the most popular methods to analyze this type of data are multiple factor analysis (MFA) and STATIS (which is an acronym for the French expression *Structuration des Tableaux à Trois Indices de la Statistique* or, approximately, "structuring three-way statistical tables" in English). MFA and STATIS are both component-based methods that extract factor structures from multitable data sets. These multitable data sets are organized as matrices with observations as rows and variables as columns that are organized in blocks. In sensory evaluation, MFA and STATIS are ideal methods when the data sets consist of products (i.e., rows) that are described by several assessors (i.e., tables) who may or may not use the same variables (i.e., columns) to evaluate these products. MFA and STATIS first weight each subtable to build an optimal combination that best represents the general pattern. Here, the definition of *optimal* differs for MFA and STATIS: MFA combines the tables by setting the spectral norm of each sub-table to unity (i.e., by dividing the elements of each subtable by its first singular value), whereas STATIS combines the tables by weighting the tables with a common pattern more and those with a rare pattern less. (Pagès[1] provides a thorough comparison between MFA and STATIS with more technical details.) From this optimal combination, MFA and STATIS then extract *orthogonal* components in a way similar to principal component analysis (PCA). There are two equivalent core techniques to extract such components: the singular value decomposition (SVD) and the eigenvalue decomposition (EVD; also called eigen-decomposition). With the SVD, MFA and STATIS decompose a grand table where all data tables are concatenated; by contrast, with the EVD, MFA and STATIS first compute the cross-product of each data table and then decompose the optimal weighted sum of these cross-products. Both techniques extract orthogonal components, where the first component explains the most variance of the data, the second component explains the second most, and so on. With either technique, MFA and STATIS extract components that best capture the variability in the data and further analyze the data structure formed by the blocks.

Another form of sensory evaluation data describes the similarity between observations based on their similarities or dissimilarities (or distance); one example of such data is obtained from a sorting task where the participants are asked to sort the observations into groups. This type of multiblock data also includes multiple data tables associated with the participants and are often represented by symmetric tables that describe pairwise relationships between the observations. To analyze such data sets, two other methods called DiSTATIS and CovSTATIS extend STATIS to extract orthogonal components from, respectively, distance and variance-covariance matrices.[2] DiSTATIS and CovSTATIS compute a weighted sum of all the tables to obtain an optimal table that best represents the general pattern. Just like in STATIS, the optimal table is computed by weighting the tables with a common pattern more and those with a rarer pattern less. Because this optimal table is symmetric with matching rows and columns, instead of using the SVD that analyzes rectangular matrices, DiSTATIS and CovSTATIS use EVD to decompose it.

Current data sets now routinely include a large number of items such as participants, observations, and attributes. Often, the analysis of these large data sets gives results that are difficult to interpret because (1) they give complex components that involve a large proportion of the items under scrutiny, and (2) the data include heterogeneous data tables which are poorly represented by a single pattern. The first issue can be solved by obtaining components that have a *simple structure*. As formalized in Thurstone,[3,4] in a simple structure, a given component is characterized by a small number of variables and each variable contributes to few (ideally one) components. To obtain such structure, early methodologists rotated the component axes[4–6]; although rotation often simplifies the components, it is limited when the data are too complex and do not have a clear construct—as is often the case with modern data. Therefore, modern statisticians[7,8] developed *sparsification* as an alternative to obtain a simple structure.

Sparsification methods originated from the general linear model framework, where they are used to eliminate negligible predictors so that the prediction from the model is more reliable. Recently, these early sparsification approaches were extended to component-based methods[9,10] because—compared to rotation— sparsification is more analytical in selecting the variables and so provides a more objective way to obtain a simple structure.[11] However, most of these methods maximize sparsification but relax other conditions such as orthogonality of loadings and factor scores. But, when the components are nonorthogonal, their interpretation becomes arduous because the components are confounded with each other and cannot be interpreted independently. Because these components now share some variance, their variances are no longer additive and the sum of these variances will over-estimate the proportion of the

variance explained by the components. To palliate this problem, Guillemot et al.[10] developed a new SVD sparsification algorithm that integrates orthogonality constraints on components and loadings. But, this algorithm has not yet been adapted to multitable methods.

In this paper, we propose a unified theoretical framework to sparsify multitable methods such as MFA, STATIS, and DiSTATIS/CovSTATIS. The algorithm is inspired by the algorithm of sparse SVD (sSVD or called the *constrained SVD* in Guillemot et al.[10]) and extended to sparsifying EVD. Although the SVD and the EVD are equivalent and can both be used to perform MFA and STATIS, their sparsification results, however, are not. In sSVD, the rows and the columns are sparsified separately; in EVD, the rows and the columns must be sparsified in pairs. To sparsify MFA and STATIS, we extended sSVD, instead of sparse EVD (sEVD), to preserve the most complete properties. For sparse MFA (sMFA) and sparse STATIS (sSTATIS), the sSVD algorithm is modified so that it not only generates orthogonal and sparse components but also sparsifies the variables in a priori defined groups (e.g., a whole subtable). Such modification aims to identify different subsets of tables that are more homogeneous. Different from sMFA and sSTATIS, DiSTATIS/CovSTATIS are sparsified by extending sEVD because the data tables they analyzed are square and symmetric with rows and columns that match each other and should, therefore, be kept or eliminated in pairs. In sparse DiSTATIS (sDiSTATIS) and sparse CovSTATIS (sCovSTATIS), the algorithm also aims to identify homogeneous subsets of tables. To demonstrate these sparse methods, we apply them to two data sets in this paper: One gives the results of a sensory evaluation experiment on food, and the other gives sensory evaluation results of experts and novices on wine tasting. We show how introducing sparsity and orthogonality constraints complements regular methods and allows users to have a rich interpretation of the data structure. We also identify the properties that are preserved or lost after imposing these constraints.

2 | BACKGROUND

2.1 | Notations and definitions

Matrices are denoted by bold uppercase letters (e.g., $\mathbf{A}$), vectors are denoted by bold lowercase letters (e.g., $\mathbf{a}$), the elements of a matrix or a vector are denoted by italic lowercase letters (e.g., a), and an integer is denoted by an italic uppercase letter (e.g., N). The matrix $\mathbf{I}$ is the identity matrix (i.e., a matrix with ones on the diagonal and zeros off the diagonal), and $\mathbf{1}$ is a single-column matrix of ones. The transpose of a matrix is denoted by the superscript $^\top$ (e.g., $\mathbf{A}^\top$).

In the analysis, the data matrix is denoted by $\mathbf{X}$. When there are multiple data tables, the total number of tables is denoted by K, and a subscript of $\mathbf{X}$ is used to denote the referred table (e.g., $\mathbf{X}_1$, $\mathbf{X}_2$,..., $\mathbf{X}_k$,..., and $\mathbf{X}_K$). For each table, the total number of rows is denoted by I, and the total number of columns is denoted by J with a subscript indicating the corresponding data table (e.g., J_k). The element stored on the ith row and jth column of the kth table will be denoted by $x_{i,j,k}$ (or $x_{i,j}$ if only one table has been mentioned). For an $I \times J$ matrix, the minimum of I and J is the largest possible *rank* (denoted L) of the matrix. For all illustrations after a matrix decomposition, the subscript ℓ is used to denote the component and sometimes also the corresponding table. For example, $\mathbf{x}_\ell$ denotes a vector $\mathbf{x}$ that corresponds to the ℓth component, and $\mathbf{x}_{k\ell}$ denotes a vector $\mathbf{x}$ that corresponds to the ℓth component of the kth table. The operator $\underset{\mathbf{x}}{\arg\max}\{f(\mathbf{x})\}$ identifies the argument $\mathbf{x}$ that maximizes the value of $f(\mathbf{x})$. Similarly, the operator $\underset{\mathbf{x}}{\arg\min}\{f(\mathbf{x})\}$ identifies the argument $\mathbf{x}$ that minimizes $f(\mathbf{x})$. The $\mathcal{L}_1$-norm of vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_1$: it is computed as the sum of the absolute values of all elements of $\mathbf{x}$; the $\mathcal{L}_2$-norm of vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_2$: it is computed as $\sqrt{\mathbf{x}^\top\mathbf{x}}$. The operator $\text{proj}_{\mathcal{C}}(\mathbf{x})$ denotes the projection of a vector $\mathbf{x}$ onto a convex set $\mathcal{C}$ and is defined as $\text{proj}_{\mathcal{C}}(\mathbf{x}) = \underset{\mathbf{y}\in\mathcal{C}}{\arg\min}\left\{\|\mathbf{x}-\mathbf{y}\|_2\right\}$.

2.2 | The singular value decomposition (SVD), eigenvalue decomposition (EVD), and their sparsifications

Two core techniques are used extensively in component-based multitable methods: The SVD and the EVD. Therefore, to sparsify these methods, we first sparsify the SVD and the EVD. For each of these techniques, we present the optimization problem that they solve and how sparsification is integrated.

2.2.1 | The SVD and sparse SVD (sSVD)

The SVD is applied to a rectangular $I \times J$ matrix $\mathbf{X}$. The optimization problem behind the SVD is the following:

$$\underset{\mathbf{P},\mathbf{\Delta},\mathbf{Q}}{\arg\min}\left\{\left\|\mathbf{X}-\mathbf{P}\mathbf{\Delta}\mathbf{Q}^{\top}\right\|_2^2\right\} \quad \text{such that} \quad \mathbf{P}^{\top}\mathbf{P}=\mathbf{Q}^{\top}\mathbf{Q}=\mathbf{I}, \tag{1}$$

where $\mathbf{P}$ (respectively $\mathbf{Q}$) is the I (respectively J)$\times L$ matrix of the left (respectively right) singular vectors $\mathbf{p}_\ell$ (respectively $\mathbf{q}_\ell$), and $\mathbf{\Delta}$ is a diagonal matrix whose diagonal stores the singular values δ_ℓs (where $\delta_1 \geq \delta_2 \geq \ldots \geq \delta_\ell \geq \ldots \geq \delta_L \geq 0$). In addition, the ℓth singular value (δ_ℓ) is the standard deviation of the elements of the ℓth component.

The optimization problem of the SVD is equivalent to iteratively solving the optimization problem for each of the ℓth dimension ($\ell \geq 1$):

$$\begin{aligned}&\underset{\mathbf{p}_\ell,\mathbf{q}_\ell}{\arg\max}\left\{\mathbf{p}_\ell^{\top}\mathbf{X}\mathbf{q}_\ell\right\}\\ &\text{such that}\begin{cases}\mathbf{p}_\ell^{\top}\mathbf{p}_\ell=\mathbf{q}_\ell^{\top}\mathbf{q}_\ell=1.\\ \mathbf{p}_\ell^{\top}\mathbf{p}_{\ell'}=\mathbf{q}_\ell^{\top}\mathbf{q}_{\ell'}=0,\ \forall \ell'<\ell, \text{ if } \ell>1.\end{cases}\end{aligned} \tag{2}$$

This optimization problem searches for $\mathbf{p}_\ell$ and $\mathbf{q}_\ell$ that maximize $\mathbf{p}_\ell^{\top}\mathbf{X}\mathbf{q}_\ell$, at which point this quantity is equal to the singular value of $\mathbf{X}$ (δ_ℓ).

To sparsify the SVD, we relax the $\mathcal{L}_2$-constraint and add two more convex non-differentiable constraints to the optimization problem;[10] specifically:

$$\begin{aligned}&\underset{\mathbf{p}_\ell,\mathbf{q}_\ell}{\arg\max}\left\{\mathbf{p}_\ell^{\top}\mathbf{X}\mathbf{q}_\ell\right\}\\ &\text{such that}\begin{cases}\mathbf{p}_\ell^{\top}\mathbf{p}_\ell\leq 1 \text{ and } \mathbf{q}_\ell^{\top}\mathbf{q}_\ell\leq 1,\\ \mathbf{p}_\ell^{\top}\mathbf{p}_{\ell'}=\mathbf{q}_\ell^{\top}\mathbf{q}_{\ell'}=0,\ \forall \ell'<\ell, \text{ if } \ell>1,\\ \|\mathbf{p}_\ell\|_1\leq s_{\mathbf{p}\ell},\\ \|\mathbf{q}_\ell\|_1\leq s_{\mathbf{q}\ell},\end{cases}\end{aligned} \tag{3}$$

where $\|\cdot\|_1$ is the $\mathcal{L}_1$-norm, and $s_{\mathbf{p}\ell}$ (possible values range $\left[1, \sqrt{I}\right]$) and $s_{\mathbf{q}\ell}$ (possible values range $\left[1, \sqrt{J}\right]$) are the sparsity parameters. Adding these constraints yields sparse singular vectors $\mathbf{p}_\ell$ and $\mathbf{q}_\ell$ with a number of zeros that is inversely related to the value of the sparsity parameters. When the singular values and vectors are sparsified, we call them the *pseudo*-singular values or the *pseudo*-singular vectors. To differentiate the pseudo-singular values from the regular ones (δ_ℓ) for later usage, we denote pseudo-singular values as $\hat{\delta}_\ell$.

This optimization problem is solved by projecting the data onto a convex set, including three convex spaces: (1) The $\mathcal{L}_2$-ball that normalizes the pseudo-singular vectors, (2) the orthogonal spaces that ensure the orthogonality of left (respectively right) pseudo-singular vectors between components, and (3) the $\mathcal{L}_1$-ball that sparsifies the pseudo-singular vectors. Geometrically, this projection can be solved by a fast and exact algorithm called the PL1L2 algorithm developed by Gloaguen et al.[12]; the convergence for these algorithms is proven in Guillemot et al.[10]

2.2.2 | The EVD and sparse EVD (sEVD)

The second method (the EVD) is applied to a square $I \times I$ matrix $\mathbf{S}$ as $\mathbf{S}$ equals the cross-product of $\mathbf{X}$ (i.e., $\mathbf{S}=\mathbf{X}\mathbf{X}^{\top}$). The optimization problem of the EVD is the following:

$$\underset{\mathbf{P},\mathbf{\Lambda}}{\arg\min}\left\{\left\|\mathbf{S}-\mathbf{P}\mathbf{\Lambda}\mathbf{P}^{\top}\right\|_2^2\right\} \quad \text{such that} \quad \mathbf{P}^{\top}\mathbf{P}=\mathbf{I}, \tag{4}$$

where $\mathbf{P}$ is the $I \times L$ matrix of eigenvectors ($\mathbf{p}_\ell$) and $\mathbf{\Lambda}$ is a diagonal matrix whose diagonal stores the eigenvalues λ_ℓs (where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_\ell \geq \ldots \geq \lambda_L \geq 0$). These eigenvectors (denoted $\mathbf{P}$) of $\mathbf{S}$ are equal to the left singular vectors

(also denoted $\mathbf{P}$) of $\mathbf{X}$, and the eigenvalues $\mathbf{\Lambda}$ of $\mathbf{S}$ are equal to the squared singular values ($\mathbf{\Delta}^2$) of $\mathbf{X}$. When all the eigenvalues are non-negative, $\mathbf{S}$ is a positive semidefinite (psd) matrix.

The optimization problem from Equation (4) is equivalent to iteratively solving—for each of the ℓth dimension ($\ell \geq 1$)—the following problem:

$$\begin{aligned} &\arg\max_{\mathbf{p}_\ell} \left\{ \mathbf{p}_\ell^\top \mathbf{S} \mathbf{p}_\ell \right\} \\ &\text{such that} \begin{cases} \mathbf{p}_\ell^\top \mathbf{p}_\ell = 1. \\ \mathbf{p}_\ell^\top \mathbf{p}_{\ell'} = 0, \ \forall \ell' < \ell, \text{ if } \ell > 1. \end{cases} \end{aligned} \tag{5}$$

This optimization problem searches for a $\mathbf{p}_\ell$ that maximizes $\mathbf{p}_\ell^\top \mathbf{S} \mathbf{p}_\ell$, at which point this quantity is equal to the eigenvalue of $\mathbf{S}$ (λ_ℓ).

To sparsify the EVD, we relax the $\mathcal{L}_2$-constraint and add another convex nondifferentiable constraint to this optimization problem:

$$\begin{aligned} &\arg\max_{\mathbf{p}_\ell} \left\{ \mathbf{p}_\ell^\top \mathbf{S} \mathbf{p}_\ell \right\} \\ &\text{such that} \begin{cases} \mathbf{p}_\ell^\top \mathbf{p}_\ell \leq 1, \\ \mathbf{p}_\ell^\top \mathbf{p}_{\ell'} = 0, \ \forall \ell' < \ell, \text{ if } \ell > 1, \\ \|\mathbf{p}_\ell\|_1 \leq s_{\mathbf{p}\ell}, \end{cases} \end{aligned} \tag{6}$$

where $\|\cdot\|_1$ is the $\mathcal{L}_1$-norm and $s_{\mathbf{p}\ell}$ is the sparsity parameter whose possible values lay in the interval $\left[1, \sqrt{I}\right]$. Adding this constraint yields a sparse eigenvector $\mathbf{p}_\ell$ whose number of zeros is inversely related to the value of this sparsity parameter. Similar to the sSVD, this optimization problem can be solved by the PL1L2 algorithm.[10,12] When the eigenvalues and eigenvectors are sparsified, we call them the *pseudo*-eigenvalues or the *pseudo*-eigenvectors. To differentiate the pseudo-eigenvalues from the regular ones (λ_ℓ) for later usage, we denote pseudo-eigenvalues as $\hat{\lambda}_\ell$.

2.2.3 | Evaluating the sparsification

The most straightforward way to evaluate the sparsity of the left and right pseudo-singular vectors is to express their numbers of zeros as a function of the sparsity parameters $s_{\mathbf{p}\ell}$ and $s_{\mathbf{q}\ell}$. But, as noted by Liu et al,[13] such a crude index of sparsity would not help a user choose an "optimal" value for the sparsity parameters. A more user friendly approach was suggested by Trendafilov et al.[14] who defined a sparsity index that combines (1) a measure of sparsity and (2) a measure of how close the reduced rank sparse matrix is to the original data matrix.

We measure sparsity with three different indices: (1) $\vartheta_{\mathbf{P}}$, the ratio of the number of zeros (denoted by $\#_0$) to the total number of coefficient in $\mathbf{P}$; (2) the counterpart $\vartheta_{\mathbf{Q}}$, the ratio of the number of zeros to the total number of coefficients in $\mathbf{Q}$; and (3) ϑ, the ratio of the number of zeros in both $\mathbf{P}$ and $\mathbf{Q}$ to the total number of coefficients in $\mathbf{P}$ and $\mathbf{Q}$:

$$\vartheta_{\mathbf{P}} = \frac{\#_0(\mathbf{P})}{I \times \ell}, \tag{7}$$

$$\vartheta_{\mathbf{Q}} = \frac{\#_0(\mathbf{Q})}{J \times \ell}, \tag{8}$$

$$\vartheta = \frac{\#_0(\mathbf{P}) + \#_0(\mathbf{Q})}{(I+J) \times \ell}. \tag{9}$$

We measure the fit, denoted $\hat{\tau}$, by computing the ratio of the sum of the first ℓ squared pseudo-singular values from the sSVD (respectively the sum of the first ℓ pseudo-eigenvalues from the sEVD) to the sum of the first ℓ squared singular values (respectively the sum of the first ℓ eigenvalues):

$$\hat{\tau} = \frac{\sum_{m=1}^{\ell} \hat{\delta}_m^2}{\sum_{m=1}^{\ell} \delta_m^2} (\text{for sSVD}) \quad \text{or} \quad \hat{\tau} = \frac{\sum_{m=1}^{\ell} \hat{\lambda}_m}{\sum_{m=1}^{\ell} \lambda_m} (\text{for sEVD}). \tag{10}$$

By combining both types of ratios (sparsity and fit), we obtain three different sparsity indices:

$$\varsigma_{\mathbf{P}} = \vartheta_{\mathbf{P}} \times \hat{\tau}, \tag{11}$$

$$\varsigma_{\mathbf{Q}} = \vartheta_{\mathbf{Q}} \times \hat{\tau}, \tag{12}$$

$$\varsigma = \vartheta \times \hat{\tau}, \tag{13}$$

where $\varsigma_{\mathbf{P}}$ is the compromise between fit and sparsity for the left pseudo-singular vectors, $\varsigma_{\mathbf{Q}}$ is the compromise between fit and sparsity for the right pseudo-singular vectors, and ς is the compromise between fit and sparsity considering both the left and the right pseudo-singular vectors. Depending on the application, the user can use one or more of these three sparsity indices as a reference to select an appropriate value for the sparsity parameters $s_{\mathbf{p}\ell}$ and $s_{\mathbf{q}\ell}$. We demonstrate the usefulness of these indices in the result section.

Figure 1 depicts the range of possible values for ς on a graph representing the ratio of zeros on the x-axis and the "fit" on the y-axis. In this figure, the result of a sparse method would be represented as a dot. According to how sparse the loadings are ("zero ratio") and how close the lower rank sparse decomposition of the data is to the original data (i.e., the "fit"), we split the graph into five zones: Zones 1 to 3 correspond to a low sparsity index because either or both "fit" and "zero ratio" are close to zero; Zone 5 corresponds to a middle ground, where a good compromise is reached between sparsity and fit; and, finally, Zone 4 corresponds to a sparsity index close to its maximum value of 1 when very few variables are selected to represent most of the information in the data. The closer the result is to the corner of Zone 4, the larger the sparsity index, and the better the sparse results represent the data.

In the following analyses, we will provide such map for each type of the analyses we run. The optimal sparsity parameters and number of dimensions will be chosen by maximizing the sparsity index.

3 | METHODS

MFA and STATIS are component-based multitable methods that extract orthogonal components from the general patterns across all tables, and DiSTATIS/CovSTATIS extends STATIS to analyze symmetric matrices. If K is the number of tables and $\mathbf{X}_k$ is the kth table, MFA and STATIS analyze the concatenated grand table $\mathbf{X}$:

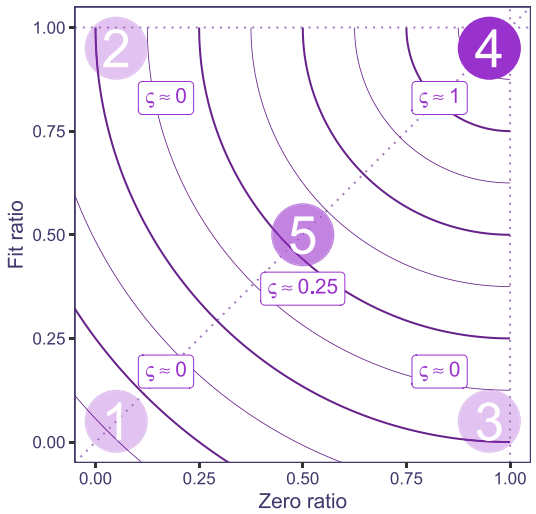

FIGURE 1 Graph representing different possible values for the sparsity index on a map of the "fit" as a function of the "zero ratio." The five zones represent five possible combinations of the two ratios, along with the corresponding value for the sparsity index ς.

$$\mathbf{X} = [\mathbf{X}_1 | \mathbf{X}_2 | \ldots | \mathbf{X}_k | \ldots | \mathbf{X}_K], \tag{14}$$

with a decomposition similar to PCA. In PCA, the variables (i.e., the columns) of the analyzed table are often centered (by subtracting the mean of a variable from all the values) and normalized (to have sums of squares or standard deviations of all variables equal to 1) to equalize the contribution of each variable. Therefore, when PCA is used to analyze the concatenated grand table, the first component is likely to be dominated by the data table with either the largest number of variables or the lowest dimensionality. To solve this problem, before building the grand table and performing PCA, MFA and STATIS/DiSTATIS both preprocessed the data tables, but in different ways. MFA normalizes each table (i.e., by dividing it by its first singular value) to give the same importance to each table so that no table dominates the first component of the grand table; STATIS and DiSTATIS/CovSTATIS normalize each table based on how common its pattern is and give more importance to tables that are closer to the common pattern. Consequently, STATIS and DiSTATIS/CovSTATIS extract components that best represent the common pattern among all data tables. In general, all these methods apply different weights to each table and perform PCA on the *weighted* grand table:

$$\begin{aligned} \tilde{\mathbf{X}} &= [\alpha_1 \mathbf{X}_1 | \alpha_2 \mathbf{X}_2 | \ldots | \alpha_k \mathbf{X}_k | \ldots | \alpha_K \mathbf{X}_K] \\ &= \left[\tilde{\mathbf{X}}_1 | \tilde{\mathbf{X}}_2 | \ldots | \tilde{\mathbf{X}}_k | \ldots | \tilde{\mathbf{X}}_K\right], \end{aligned} \tag{15}$$

where α_k is the weight applied to the kth table. Because the weights in MFA and STATIS/DiSTATIS/CovSTATIS have different aims, we denote them differently in the following sections. The weights for MFA are denoted by α_k, and the weights for STATIS/DiSTATIS/CovSTATIS are denoted by β_k.

3.1 | Multiple factor analysis (MFA) and sparse MFA (sMFA): analyzing the average pattern across tables

In MFA, the tables are preprocessed by normalizing each table by its first singular value. Because the first singular value of a table gives the standard deviation of its first component, the first normalization step is similar to performing a Z-score normalization (i.e., each table is divided by the standard deviation of its first component). This step ensures that each table has a first singular value equal to one, and so no table can dominate the analysis only because of its larger first singular value. The procedure of MFA is illustrated in Figure 2A.

3.1.1 | Theory

Formally, MFA consists of three steps (for details see, e.g., Abdi et al.[15] and Pagés[16]). First, MFA computes the SVD for each table $\mathbf{X}_k$:

$$\mathbf{X}_k = \mathbf{U}_k \boldsymbol{\Gamma}_k \mathbf{V}_k^\top \text{ under the constraints } \mathbf{U}_k^\top \mathbf{U}_k = \mathbf{V}_k^\top \mathbf{V}_k = \mathbf{I}. \tag{16}$$

Next, MFA extracts the first singular value (denoted by γ_{k1}) of the all K tables and weights them by α_k, which is the inverse of γ_{k1}:

$$\alpha_k = \gamma_{k,1}^{-1}. \tag{17}$$

The weighted tables are denoted as $\tilde{\mathbf{X}}_k$, where

$$\tilde{\mathbf{X}}_k = \alpha_k \mathbf{X}_k. \tag{18}$$

The weighted tables are then concatenated to create the grand table (denoted by $\tilde{\mathbf{X}}$):

$$\begin{aligned} \tilde{\mathbf{X}} &= [\alpha_1 \mathbf{X}_1 | \alpha_2 \mathbf{X}_2 | \ldots | \alpha_k \mathbf{X}_k | \ldots | \alpha_K \mathbf{X}_K] \\ &= \left[\tilde{\mathbf{X}}_1 | \tilde{\mathbf{X}}_2 | \ldots | \tilde{\mathbf{X}}_k | \ldots | \tilde{\mathbf{X}}_K\right]. \end{aligned} \tag{19}$$

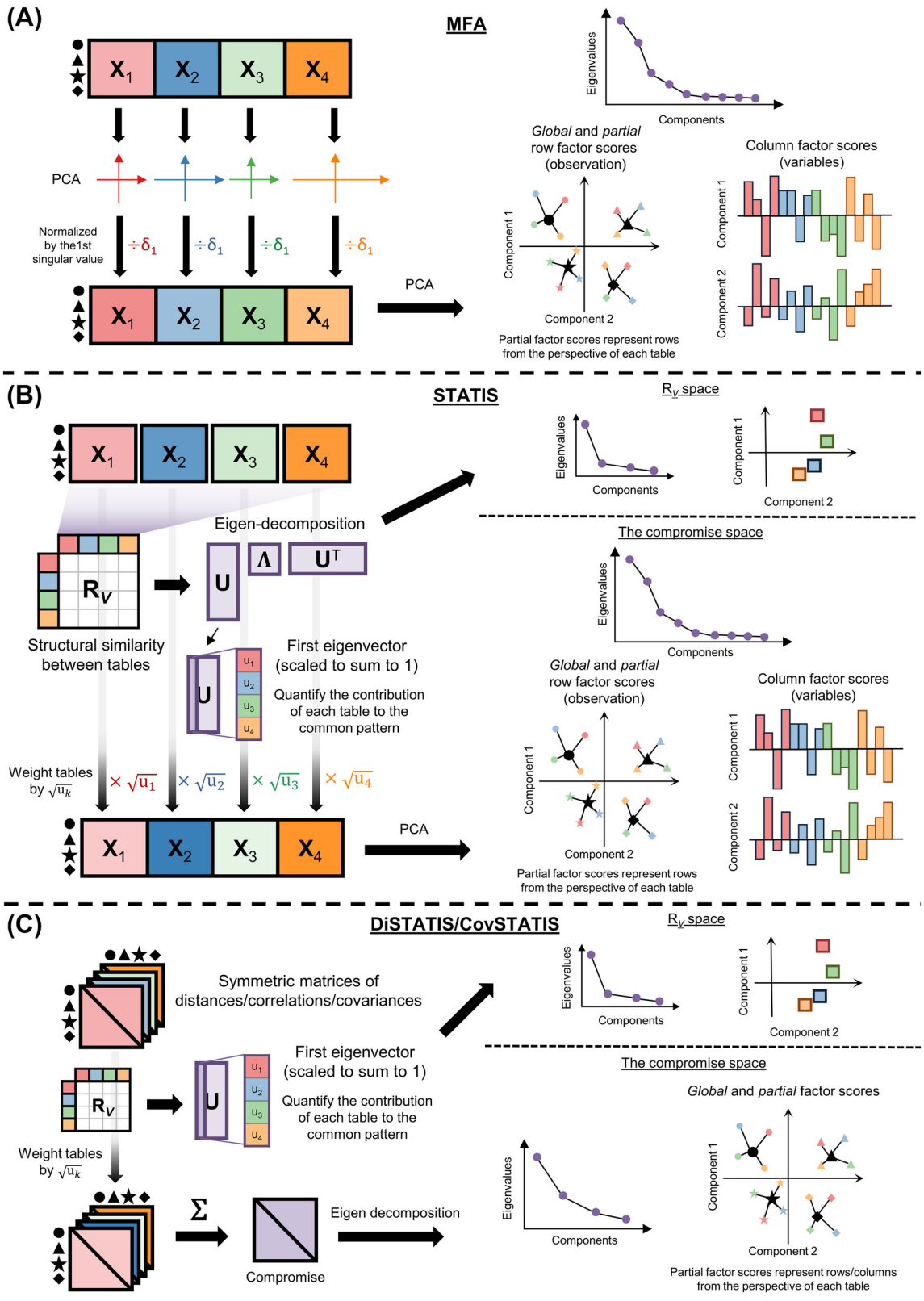

FIGURE 2 This figure illustrates the procedures of MFA (A), STATIS (B), and DiSTATIS/CovSTATIS (C).

Finally, this grand table is decomposed by the SVD:

$$\tilde{\mathbf{X}} = \mathbf{P}\Delta\mathbf{Q}^{\top}, \text{ where } \mathbf{P}^{\top}\mathbf{P} = \mathbf{Q}^{\top}\mathbf{Q} = \mathbf{I}. \tag{20}$$

Here, $\mathbf{Q}$ is structured in blocks in the same way as $\tilde{\mathbf{X}}$:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 \\ \mathbf{Q}_2 \\ \vdots \\ \mathbf{Q}_k \\ \vdots \\ \mathbf{Q}_K \end{bmatrix}. \tag{21}$$

Note that, in general, the constraint in Equation (20) does not apply when only one table is considered:

$$\mathbf{Q}_k^\top \mathbf{Q}_k \neq \mathbf{I}. \tag{22}$$

As an SVD-based method, MFA projects the rows (which often correspond to observations) onto components as *row factor scores* and projects columns (which often correspond to variables) onto components as *column factor scores*. In multitable methods, there are two types of row factor scores: the *global* and the *partial* factor scores. The *global* factor scores (denoted as $\mathbf{F}$) are computed as

$$\mathbf{F} = \mathbf{P}\boldsymbol{\Delta} = \tilde{\mathbf{X}}\mathbf{Q}. \tag{23}$$

Because $\mathbf{Q}$ includes variables of all data tables (cf. Equation 21), the global factor scores describe the observations from the viewpoint of the whole grand table. By contrast, the *partial* factor scores (denoted $\mathbf{F}_k$) correspond to the each of the K tables and are computed as

$$\mathbf{F}_k = K\tilde{\mathbf{X}}_k\mathbf{Q}_k. \tag{24}$$

These scores represent the rows on a given component from the perspectives of individual tables.

The set of partial factor scores of each observation illustrates how different data tables contribute to the global factor scores of an observation. Therefore, the *barycenters* (i.e., the means) of these sets of partial factor scores are the global factor scores (cf. Equation 23):

$$\begin{aligned} \frac{1}{K}\sum_{k=1}^{K}\mathbf{F}_k &= \frac{1}{K}\sum_{k=1}^{K}K\tilde{\mathbf{X}}_k\mathbf{Q}_k \\ &= \frac{1}{K}K\sum_{k=1}^{K}\tilde{\mathbf{X}}_k\mathbf{Q}_k \\ &= \sum_{k=1}^{K}\tilde{\mathbf{X}}_k\mathbf{Q}_k \\ &= \tilde{\mathbf{X}}\mathbf{Q} = \mathbf{F}. \end{aligned} \tag{25}$$

Finally, the column factor scores $\mathbf{G}$ are computed as

$$\mathbf{G} = \alpha_k^{-1}\mathbf{Q}\boldsymbol{\Delta}. \tag{26}$$

3.1.2 | Sparse MFA (sMFA): using sparse SVD with group constraints

To sparsify MFA, we sparsified the SVD of MFA by solving the iterative optimization problem of the sSVD (cf. Equation 3). In sMFA, we modified the constraints of the sSVD by replacing the $\mathcal{L}_1$-constraints with the $\mathcal{L}_\mathcal{G}$-constraints so that the variables that belong to the same table will be eliminated or kept at the same time. Formally, the optimization problem becomes

$$\arg\max_{\mathbf{p}_\ell,\mathbf{q}_\ell}\left\{\mathbf{p}_\ell^\top\tilde{\mathbf{X}}\mathbf{q}_\ell\right\}$$
$$\text{such that}\begin{cases}\mathbf{p}_\ell^\top\mathbf{p}_\ell\le 1 \text{ and } \mathbf{q}_\ell^\top\mathbf{q}_\ell\le 1,\\ \mathbf{p}_\ell^\top\mathbf{p}_{\ell'}=\mathbf{q}_\ell^\top\mathbf{q}_{\ell'}=0,\ \forall\ell'<\ell,\ \text{if } \ell>1,\\ \|\mathbf{p}_\ell\|_1\le s_{\mathbf{p}\ell},\\ \|\mathbf{q}_\ell\|_{\mathcal{G}}\le s_{\mathbf{q}\ell},\end{cases}\tag{27}$$

which searches for $\mathbf{p}_\ell$ and $\mathbf{q}_\ell$ that maximize $\mathbf{p}_\ell^\top\tilde{\mathbf{X}}\mathbf{q}_\ell$ under sparsity constraints, at which point this quantity is equal to the the pseudo-singular value of the grand table ($\hat{\delta}_\ell$). In this equation, the group norm is defined as "the (1,2)-group norm" in van den Berg et al.[17] where $\|\mathbf{x}\|_{\mathcal{G}}=\sum_{g=1}^{G}\left\|\mathbf{x}_{\iota_g}\right\|_2$, where $\mathbf{x}_{\iota_g}$ are all ι_g elements of the vector $\mathbf{x}$ that belong to the gth group. Here, we call this group norm the $\mathcal{L}_{\mathcal{G}}$-norm.

With the PL1L2 algorithm, this optimization problem can be solved by projecting the data onto three convex spaces: (1) the $\mathcal{L}_2$-ball that normalizes the pseudo-singular vectors, (2) the orthogonal spaces that ensure the orthogonality between left (respectively right) pseudo-singular vectors of different components, and (3) the $\mathcal{L}_{\mathcal{G}}$-ball that is used instead of the $\mathcal{L}_1$-ball to sparsify the variables in groups. The optimal solution of the optimization problem is identified based on the sparsity index.

The derived $\mathbf{p}_\ell$ and $\mathbf{q}_\ell$ are then stored into the sparsified $\mathbf{P}$ and $\mathbf{Q}$ matrices, where

$$\begin{aligned}\mathbf{P}&=[\mathbf{p}_1|\mathbf{p}_2|...|\mathbf{p}_\ell|...|\mathbf{p}_L]\\ \mathbf{Q}&=[\mathbf{q}_1|\mathbf{q}_2|...|\mathbf{q}_\ell|...|\mathbf{q}_L].\end{aligned}\tag{28}$$

With the sparsified $\mathbf{Q}$, the global and partial row factor scores are computed using Equations (23) and (24), and the column factor scores are computed using Equation (26).

3.2 | STATIS and sparse STATIS (sSTATIS): analyzing the common pattern across tables

STATIS—another SVD-based method—adopts a strategy different from MFA to pre-process the data tables. In STATIS, the tables with a common pattern are weighted more and the tables with a rare pattern are weighted less so that the extracted components best represent the general structure of data tables. Specifically, STATIS uses the *R_V coefficient*[18] to quantify the structural similarity between each pair of data tables. The procedure of STATIS is illustrated in Figure 2B.

3.2.1 | Theory

Formally, STATIS comprises four steps. First, STATIS computes an $I\times I$ symmetric cross-product matrix $\mathbf{S}_k$ from each $I\times J$ data table $\mathbf{X}_k$ as

$$\mathbf{S}_k=\mathbf{X}_k\mathbf{X}_k^\top\tag{29}$$

where both the rows and columns of $\mathbf{S}_k$ represent the I observations in $\mathbf{X}_k$. The sums of squares of these observations are stored in the diagonal elements of $\mathbf{S}_k$, and the cross-product of pairs of observations are stored in the off-diagonal elements of $\mathbf{S}_k$. Therefore, $\mathbf{S}_k$ measures how these observations are associated according to the variables of $\mathbf{X}_k$.

In the second step, the structural similarity between pairs of data tables (e.g., $\mathbf{S}_k$ and $\mathbf{S}_{k'}$) is computed using Escoufier's vector correlation denoted by *R_V coefficient*[18] defined as

$$R_{\mathrm{V}}=\frac{\operatorname{trace}(\mathbf{S}_k^\top\mathbf{S}_{k'})}{\sqrt{\operatorname{trace}(\mathbf{S}_k^\top\mathbf{S}_k)\operatorname{trace}(\mathbf{S}_{k'}^\top\mathbf{S}_{k'})}}=\frac{\sum_i^I\sum_j^J s_{i,j,k}s_{i,j,k'}}{\sqrt{\left(\sum_i^I\sum_j^J s_{i,j,k}^2\right)\left(\sum_i^I\sum_j^J s_{i,j,k'}^2\right)}}.\tag{30}$$

The R_V coefficient is obviously similar to a coefficient of correlation (cf. Equation 30); but—because $\mathbf{S}_k$ is a quadratic form of $\mathbf{X}_k$ (cf. Equation 29)—the R_V coefficient is better interpreted as a *squared* coefficient of correlation and ranges between 0 and 1. In contrast to the coefficient of correlation, the R_V coefficient between two tables measures the similarity between their data patterns beyond expansion and rotation.

The R_V coefficients of all pairs of tables are stored in a symmetric matrix $\mathbf{C}$, where $c_{k,k'}$ is the R_V coefficient between $\mathbf{S}_k$ and $\mathbf{S}_{k'}$.

The third step is to eigendecompose $\mathbf{C}$ as

$$\mathbf{C} = \mathbf{U}\boldsymbol{\Omega}\mathbf{U}^\top \quad \text{such that} \quad \mathbf{U}^\top\mathbf{U} = \mathbf{I}, \tag{31}$$

where $\boldsymbol{\Omega}$ is an $R \times R$ diagonal matrix of the eigenvalues with R denoting the rank of $\mathbf{C}$, and $\mathbf{U}$ is a $K \times R$ matrix of the eigenvectors of $\mathbf{C}$. Because $\mathbf{C}$ gives the similarities of all pairs of data tables, the first component of $\mathbf{C}$ is the best representation of the common pattern across tables, and thus the first eigenvector ($\mathbf{u}_1$) measures how similar each data table is to this common pattern. These similarities are then scaled to sum to one and obtained as the weight (denoted by β_k) of each table:

$$\beta_k = \sqrt{\frac{u_{k1}}{\sum_{k=1}^{K} u_{k1}}} = \left(u_{k1}(\mathbf{1}^\top\mathbf{u}_1)^{-1}\right)^{\frac{1}{2}}, \tag{32}$$

where $\mathbf{u}_{k1}$ is the k th element of $\mathbf{u}_1$ and corresponds to the k th table. Similar to MFA, each $\mathbf{X}_k$ is normalized with a multiplication by β_k:

$$\tilde{\mathbf{X}}_k = \beta_k\mathbf{X}_k, \tag{33}$$

and concatenated to create the grand table:

$$\begin{aligned}\tilde{\mathbf{X}} &= [\beta_1\mathbf{X}_1|\beta_2\mathbf{X}_2|\ldots|\beta_k\mathbf{X}_k|\ldots|\beta_K\mathbf{X}_K] \\ &= \left[\tilde{\mathbf{X}}_1|\tilde{\mathbf{X}}_2|\ldots|\tilde{\mathbf{X}}_k|\ldots|\tilde{\mathbf{X}}_K\right].\end{aligned} \tag{34}$$

Finally, the grand table is decomposed by the SVD:

$$\tilde{\mathbf{X}} = \mathbf{P}\Delta\mathbf{Q}^\top, \text{where } \mathbf{P}^\top\mathbf{P} = \mathbf{Q}^\top\mathbf{Q} = \mathbf{I}. \tag{35}$$

As in MFA, $\mathbf{Q}$ is structured in blocks in the same way as $\tilde{\mathbf{X}}$ (cf. Equation 21) and the constraint on $\mathbf{Q}$, again, does not apply when only one table is considered:

$$\mathbf{Q}_k^\top\mathbf{Q}_k \neq \mathbf{I}. \tag{36}$$

Similar to MFA, STATIS computes the global factor scores as

$$\mathbf{F} = \mathbf{P}\boldsymbol{\Delta} = \tilde{\mathbf{X}}\mathbf{Q} \tag{37}$$

to represent the observations from the viewpoint of the whole grand table, and computes partial factor scores to represent the observations from the perspectives of different tables. In STATIS, the partial factor scores $\mathbf{F}_k$ are computed by inversely weighting $\tilde{\mathbf{X}}_k$ and $\mathbf{Q}_k$ by β_k:

$$\mathbf{F}_k = \beta_k^{-1}\tilde{\mathbf{X}}_k \cdot \beta_k^{-1}\mathbf{Q}_k = \beta_k^{-2}\tilde{\mathbf{X}}_k\mathbf{Q}_k. \tag{38}$$

Consequently, the weighted mean of each set of partial factor scores gives the corresponding global factor score computed from the grand table:

$$\sum_{k=1}^{K} \beta_k^2 \mathbf{F}_k = \sum_{k=1}^{K} \beta_k^2 \beta_k^{-2} \tilde{\mathbf{X}}_k \mathbf{Q}_k = \tilde{\mathbf{X}}\mathbf{Q} = \mathbf{F}. \tag{39}$$

In addition, the column factor scores are computed as

$$\mathbf{G}_k = \beta_k^{-1} \mathbf{Q}_k \boldsymbol{\Delta}. \tag{40}$$

3.2.2 | Sparse STATIS (sSTATIS): using the sparse SVD with group constraints

The sparsification of STATIS is similar to the sparsification of MFA, where we modified the constraints of the sSVD by replacing the $\mathcal{L}_1$-constraints with the $\mathcal{L}_{\mathcal{G}}$-constraints so that the variables that belong to the same table will be eliminated or kept together. Formally, the optimization problem becomes

$$\begin{aligned} &\arg\max_{\mathbf{p}_\ell, \mathbf{q}_\ell} \left\{ \mathbf{p}_\ell^\top \tilde{\mathbf{X}} \mathbf{q}_\ell \right\} \\ &\text{such that} \begin{cases} \mathbf{p}_\ell^\top \mathbf{p}_\ell \leq 1 \text{ and } \mathbf{q}_\ell^\top \mathbf{q}_\ell \leq 1, \\ \mathbf{p}_\ell^\top \mathbf{p}_{\ell'} = \mathbf{q}_\ell^\top \mathbf{q}_{\ell'} = 0, \ \forall \ell' < \ell, \ \text{if } \ell > 1, \\ \|\mathbf{p}_\ell\|_1 \leq s_{\mathbf{p}\ell}, \\ \|\mathbf{q}_\ell\|_{\mathcal{G}} \leq s_{\mathbf{q}\ell}, \end{cases} \end{aligned} \tag{41}$$

which searches for $\mathbf{p}_\ell$ and $\mathbf{q}_\ell$ that maximize $\mathbf{p}_\ell^\top \tilde{\mathbf{X}} \mathbf{q}_\ell$, under sparsity constraints, at which point this quantity is equal to the pseudo-singular value of $\tilde{\mathbf{X}}$ ($\hat{\delta}_\ell$).

Just like in sMFA, this optimization problem can be solved by projecting the data onto 3 convex spaces with the PL1L2 algorithm. These three spaces include (1) the $\mathcal{L}_2$-ball that normalizes the pseudo-singular vectors, (2) the orthogonal spaces that ensure the orthogonality between left (respectively right) pseudo-singular vectors of different components, and (3) the $\mathcal{L}_{\mathcal{G}}$-ball that replaces the original $\mathcal{L}_1$-ball from the sSVD to further sparsify the variables in groups. Similar to sMFA, the optimal solution of the optimization problem of sSTATIS is also chosen based on the sparsity index.

The derived $\mathbf{p}_\ell$ and $\mathbf{q}_\ell$ are then stored into the sparsified $\mathbf{P}$ and $\mathbf{Q}$ matrices, and the global and partial factor scores are computed using Equations (37), (38), and (40).

3.2.3 | Alternative way to sparsify STATIS: sparsifying the R_V matrix

In addition to sparsifying corresponding variables in groups, an alternative way to sparsify the tables in sSTATIS is to sparsify the EVD of the R_V matrix $\mathbf{C}$ (cf. Equation 31). Because the eigenvectors of $\mathbf{C}(\mathbf{U})$ is a $K \times R$ matrix whose K rows correspond to the K tables, sparsifying $\mathbf{U}$ will sparsify the tables. Here, we used the iterative algorithm of sEVD with the following optimization problem:

$$\begin{aligned} &\arg\max_{\mathbf{u}_r} \left\{ \mathbf{u}_r^\top \mathbf{C} \mathbf{u}_r \right\} \\ &\text{such that} \begin{cases} \mathbf{u}_r^\top \mathbf{u}_r \leq 1, \\ \mathbf{u}_r^\top \mathbf{u}_{r'} = 0, \ \forall r' < r, \ \text{if } r > 1, \\ \|\mathbf{u}_r\|_1 \leq s_{\mathbf{u}r}, \end{cases} \end{aligned} \tag{42}$$

which searches for $\mathbf{u}_r$ that maximizes $\mathbf{u}_r^\top \mathbf{C} \mathbf{u}_r$ under sparsity constraints, at which point this quantity is equal to the pseudo-eigenvalue of $\mathbf{C}(\hat{\omega}_r)$. Here, $\mathbf{u}_r$ is the rth pseudo-eigenvector of $\mathbf{C}$, and $s_{\mathbf{u}r}$ is the sparsity parameter. This optimization problem is equivalent to projecting the data onto three convex spaces: (1) The $\mathcal{L}_2$-ball that normalizes the pseudo-singular vectors, (2) the orthogonal spaces that ensure the orthogonality between pseudo-singular eigenvectors of different components, and (3) the $\mathcal{L}_1$-ball that sparsifies $\mathbf{u}_r$.

In this version of sSTATIS, the sparse $\mathbf{u}_r$ comprises zero and nonzero values. Because the grand table are generated by weighting and concatenating data tables with weights obtained from $\mathbf{u}_r$, only the nonzero elements are used to compute the the grand table. So, instead of using only the first component of $\mathbf{U}$ to build the grand table, we consider multiple components of $\mathbf{U}$ and build separate subspaces of the grand table until all tables are used. However, to ensure that $\mathbf{S}$ is psd, the β_k weights (and therefore the elements of $\mathbf{u}_r$) should all be non-negative. Because the R_V matrix $\mathbf{C}$ only includes non-negative values, its first eigenvector (or pseudo-eigenvector) will have all elements with the same sign (according to the Perron-Frobenius theorem) which can be chosen to be positive—a property that makes this first eigenvector a perfect candidate to obtain non-negative weights. But, the components beyond the first component are, in general, composed of both negative and positive elements and therefore cannot be used to compute a grand table. To palliate this problem, we impose a non-negative constraint that ensure that the elements of $\mathbf{u}_r$ are always non-negative. To incorporate this constraint, we propose a modified sEVD algorithm that searches for pseudo-eigenvectors that are sparse, orthogonal, and positive. Appendix A details the proof that such a non-negative constraint can be imposed by partitioning the R_V matrix with sEVD to define *subspaces*. Formally, the first subspace is obtained by applying a sEVD to the R_V matrix and computing the weights from its first pseudo-eigenvector. Because some of the vector elements (and therefore their weights) are null, this subspace can also be seen as being defined by the subset of tables with non-zero weights. To obtain the next subspace, we apply the same algorithm to the sub-matrix of the R_V matrix that corresponds to the complement of Subspace 1. By iterating the same algorithm until the matrix is fully decomposed, we obtain all subspaces with a set of complementary pseudo-eigenvectors. The complete procedure is detailed in Algorithm 1.

Algorithm 1: How to sparsify the R_V matrix.

Data: $\mathbf{C}$, R
Result: A set of R positive sparse pseudo-eigenvectors of $\mathbf{C}$
$\mathbf{U} \leftarrow \emptyset$;
for $r = 1, \ldots, R$ **do**
  $\mathbf{u} \leftarrow$ sEVD($\mathbf{C}$) with optimal sparsity;
  $\mathbf{U} \leftarrow [\,\mathbf{U} \mid \mathbf{u}\,]$;
  Restrict $\mathbf{C}$ to the null coefficients in $\mathbf{u}$;
end

Here, the optimal sparsity parameter for each of the rth component $\mathbf{u}_r$ (i.e., $s_{\mathbf{u}r}$) is determined by the sparsity index. Note that Algorithm 1 requires the user to specify the number of sparse pseudo-eigenvectors R to extract from the R_V matrix. For the two examples that we analyzed, we adopted a data-driven approach to this problem: We chose R based on the prior knowledge that we had about the experimental design, and we further confirmed this choice with an additional hierarchical clustering analysis on the R_V matrix.

Next, the weights for the K tables β_{kr} are derived from $\mathbf{u}_r$ (as in Equation 32) to construct the rth subspace. Finally, the grand tables of different subspaces are constructed as in Equation (34) and decomposed by PCA as in Equation (35). For each subspace, the global and the partial factor scores are obtained by Equations (37) and (38), and the column factor scores are computed by Equation (40).

3.3 | Alternative ways to perform MFA and STATIS

Additionally to the procedures described in Sections 3.1.1 and 3.2.1, MFA and STATIS can also be performed by decomposing the cross-product matrices of all $\mathbf{X}_k$ denoted by $\mathbf{S}_k$, where

$$\mathbf{S}_k = \mathbf{X}_k \mathbf{X}_k^\top . \tag{43}$$

This version of MFA or STATIS does not decompose a concatenated grand table, but decomposes an optimal linear combination (i.e., a weighted sum) of the $\mathbf{S}_k$ matrices, which is also called the *compromise* (denoted by $\mathbf{S}_+$). Here, the weights for the tables are the same as in Equations (17) and (32).

In this version of MFA, the weight for $\mathbf{S}_k$ (here denoted by α_k^*) is computed as the squared α_k, which is also the inverse of the first eigenvalue of each table:

$$\alpha_k^* = \alpha_k^2 = \left(\gamma_{k,1}^{-1}\right)^2 = \gamma_{k,1}^{-2}, \tag{44}$$

and the compromise is computed as

$$\mathbf{S}_+ = \sum_{k=1}^{K} \alpha_k^* \mathbf{S}_k. \tag{45}$$

Similarly, in this approach of STATIS, the weights for $\mathbf{S}_k$ (here denoted by β_k^*) are computed as the squared β_k:

$$\beta_k^* = \beta_k^2 = \left(\sqrt{\frac{u_{k1}}{\sum_{k=1}^{K} u_{k1}}}\right)^2 = \frac{u_{k1}}{\sum_{k=1}^{K} u_{k1}} = u_{k1}\left(\mathbf{1}^\top \mathbf{u}_1\right)^{-1}. \tag{46}$$

Therefore,

$$\mathbf{S}_+ = \sum_{k=1}^{K} \beta_k^* \mathbf{S}_k. \tag{47}$$

Next, these compromise matrices are decomposed by EVD respectively for MFA and STATIS:

$$\mathbf{S}_+ = \mathbf{P}^* \mathbf{\Lambda} \mathbf{P}^{*\top} \quad \text{such that} \quad \mathbf{P}^{*\top} \mathbf{P}^* = \mathbf{I}. \tag{48}$$

Since this EVD decomposes the compromise, the space spanned by the extracted components is often called the *compromise space*. The global factor scores and the partial factor scores are then computed as

$$\mathbf{F} = \mathbf{S}_+ \mathbf{P}^* \mathbf{\Lambda}^{-\frac{1}{2}} \quad \text{and} \quad \mathbf{F}_k = \mathbf{S}_k \mathbf{P}^* \mathbf{\Lambda}^{-\frac{1}{2}}, \tag{49}$$

and the column factor scores are computed with the non-weighted grand table $\mathbf{X}$ as

$$\mathbf{G} = \mathbf{X}^\top \mathbf{P}^*. \tag{50}$$

These factor scores are the same as those derived from the grand table approach. This specific version of STATIS can be applied to analyze other symmetric matrices such as dissimilarity matrices (e.g., distance matrices) or similarity matrices (e.g., correlation or covariance matrices). When STATIS is applied to analyze distance matrices, it is called DiSTATIS; when applied to covariance matrices, it is called CovSTATIS.[2]

3.4 | DiSTATIS, CovSTATIS, and their sparse versions: STATIS on symmetric matrices

3.4.1 | Theory

DiSTATIS and CovSTATIS[2] extend the version of STATIS that analyzes the cross-product matrices to analyze distance or covariance matrices. The key difference between DiSTATIS and CovSTATIS is that the covariance matrices that CovSTATIS analyzes are psd matrices, but the distance matrices in DiSTATIS are not. Therefore, DiSTATIS requires an additional step to first transform the distance matrices into (psd) cross-product matrices. The procedure of DiSTATIS/CovSTATIS is illustrated in Figure 2C.

Formally, given K $I \times I$ distance matrices $\mathbf{D}_k$, we first define a *centering* matrix as

$$\boldsymbol{\Xi} = \mathbf{I} - \mathbf{1}\mathbf{m}^\top, \tag{51}$$

where $\mathbf{I}$ is the $I \times I$ identity matrix, $\mathbf{1}$ is a $K \times 1$ vector of 1s, and $\mathbf{m}$ is a mass vector defined as a $K \times 1$ vector with all elements equal $\frac{1}{K}$. We then transform each distance matrix $\mathbf{D}_k$ into a cross product matrix denoted $\mathbf{S}_k$ and obtained as

$$\mathbf{S}_k = -\frac{1}{2}\boldsymbol{\Xi}\mathbf{D}_k\boldsymbol{\Xi}^\top. \tag{52}$$

Because $\mathbf{D}_k$ is pre- and post-multiplied by the centering matrix $\boldsymbol{\Xi}$,$\mathbf{S}_k$ is said to be *double-centered*. Note that, when $\mathbf{S}_k$ is a squared Euclidean distance matrix, $\mathbf{S}_k$ is psd.

For CovSTATIS, $\tilde{\mathbf{S}}_k$ denotes a correlation or covariance matrix (which is already a psd matrix). In most cases this matrix is also doubled centered to obtained $\mathbf{S}_k$ as

$$\mathbf{S}_k = \frac{1}{2}\boldsymbol{\Xi}\tilde{\mathbf{S}}_k\boldsymbol{\Xi}^\top \tag{53}$$

(note that, compared to Equation 52, there is no minus sign). If the double centering step is not considered of interest, then $\mathbf{S}_k = \tilde{\mathbf{S}}_k$ for CovSTATIS.

From $\mathbf{S}_k$, the weights for the tables are computed as in Equation (46), and the compromise matrix is computed as in Equation (47). Finally, DiSTATIS and CovSTATIS decompose the data according to Equation (48). Because DiSTATIS and CovSTATIS analyze symmetric matrices, the factor scores for rows and columns are identical. The global factor scores that represent the variables (i.e., the rows/columns) in the compromise space are computed as

$$\mathbf{F} = \mathbf{S}_+\mathbf{P}^*\boldsymbol{\Lambda}^{-\frac{1}{2}}, \tag{54}$$

and the partial factor scores that represent the variables in the same space but from the perspective of individual tables are computed as

$$\mathbf{F}_k = \mathbf{S}_k\mathbf{P}^*\boldsymbol{\Lambda}^{-\frac{1}{2}}. \tag{55}$$

3.4.2 | Sparse DiSTATIS/CovSTATIS (sDiSTATIS/sCovSTATIS): using sEVD

Just like in sSTATIS, there are also two ways to sparsify DiSTATIS/CovSTATIS: (1) sparsifying the compromise or (2) sparsifying the R_V coefficient matrix. To sparsify the compromise, we used sEVD to decompose the compromise matrix $\mathbf{S}_+$. Formally, the optimization problem becomes

$$\begin{aligned} &\arg\max_{\mathbf{p}_\ell^*} \left\{ \mathbf{p}_\ell^{*\top} \mathbf{S}_+ \mathbf{p}_\ell^* \right\} \\ &\text{such that} \begin{cases} \mathbf{p}_\ell^{*\top} \mathbf{p}_\ell^* \leq 1, \\ \mathbf{p}_\ell^{*\top} \mathbf{p}_{\ell'}^* = 0, \ \forall \ell' < \ell, \text{ if } \ell > 1, \\ \|\mathbf{p}_\ell^*\|_1 \leq s_{\mathbf{p}^*,\ell}, \end{cases} \end{aligned} \tag{56}$$

which searches for $\mathbf{p}_\ell^*$ that maximizes $\mathbf{p}_\ell^{*\top}\mathbf{S}_+\mathbf{p}_\ell^*$ under sparsity constraints, at which point this quantity equals the pseudo-eigenvalue of $\mathbf{S}_+$ ($\hat{\lambda}_\ell$). Here, $\|\cdot\|_1$ is the $\mathcal{L}_1$-norm and $s_{\mathbf{p}^*\ell}$ is the sparsity parameter that is linked to yield sparse pseudo-eigenvector $\mathbf{p}_\ell^*$ with a number of zeros. This optimization problem can be solved by projecting the compromise $\mathbf{S}_+$ onto the 3 convex spaces with the PL1L2 algorithm. These three spaces include (1) the $\mathcal{L}_2$-ball that normalizes the pseudo-eigenvectors, (2) the orthogonal spaces that ensure the orthogonality between pseudo-eigenvectors of different components, and (3) the $\mathcal{L}_1$-ball that sparsifies the variables. The optimal solution of the optimization problem of sDiSTATIS/sCovSTATIS is also decided based on the sparsity index.

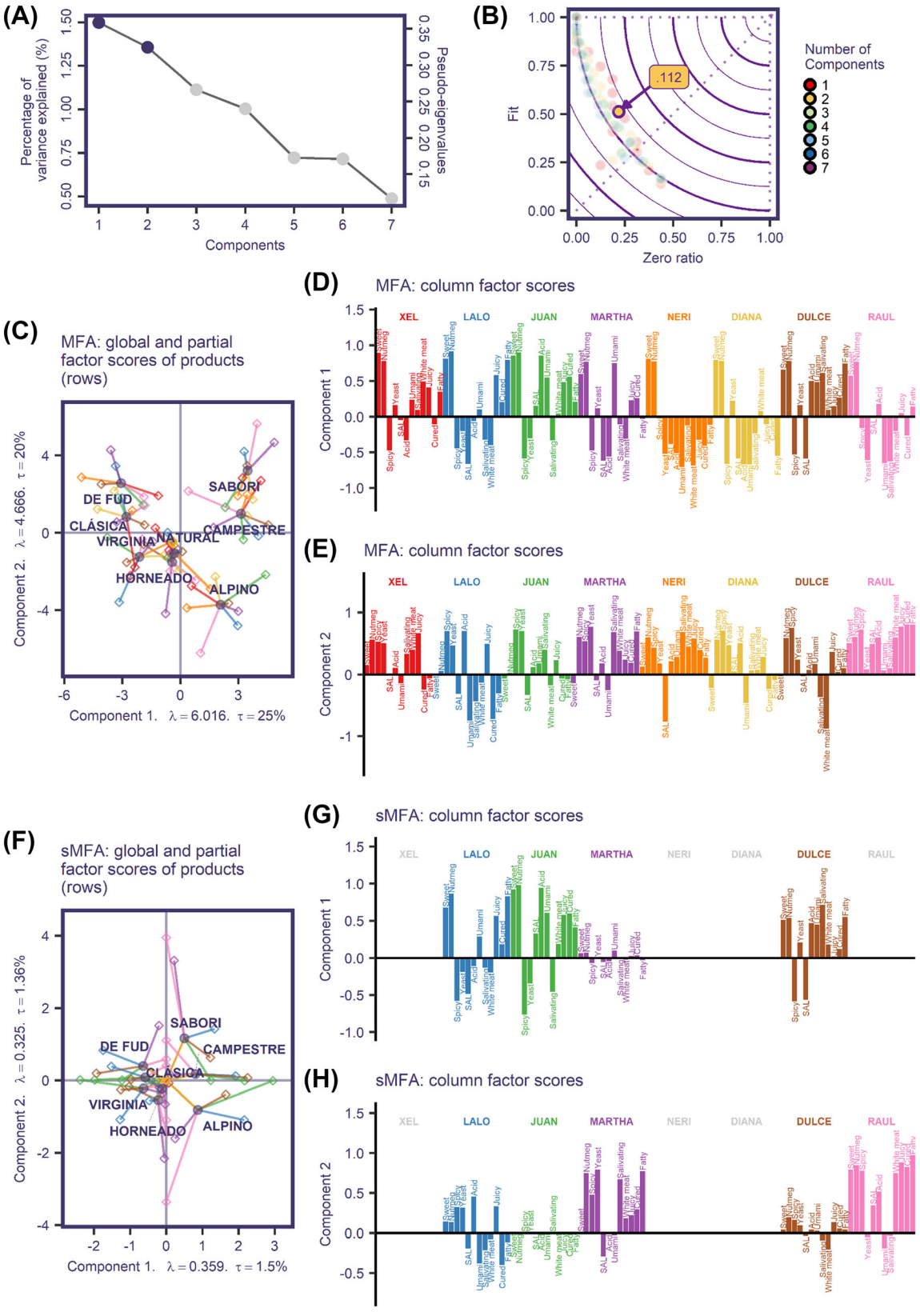

FIGURE 3 Legend on next page.

FIGURE 3 Results from MFA and sMFA. SMFA identifies a two-component solution (scree plot shown in A with the identified components colored in purple) with a maximum sparsity index of .112 indicated in B. The row factor scores of MFA, which are shown in C, and the row factor scores of sMFA, which are shown in F, have a similar pattern that separates the products with sMFA illustrating a more compact space. In these two figures, the solid points illustrate the global factor scores that represent the products in the component space, and the hollow diamonds connected to these global factor scores are the partial factor scores that represent how each product is seen from the perspective of different raters. The first two components from MFA and sMFA are associated to the pattern of attributes in the column factor scores shown respectively in barplots D–E and G–H. In these figures, the bars are organized and colored according to the raters. Each bar represents an attribute and, for each block, the bars are presented in the same order: sweet, nutmeg, spicy, yeast, salty, acid, umami, salivating, white meat, juicy, cured, and fatty. In the sMFA barplots G–H, the sparsified tables are grayed out; some non-zero loadings are too small to be seen on the same scale, but the colored variable names are still presented along the axis. The column factor scores from sMFA also identify Lalo, Juan, Martha, and Dulce for Component 1, and Lalo, Martha, Dulce, and Raul for Component 2. λ denotes the (pseudo-)eigenvalues and τ denotes the proportion of explained variance.

The derived $\mathbf{p}_\ell^*$ are then stored in the sparsified $I \times L$ matrix $\mathbf{P}^*$, and the global factor scores are computed as in Equation (54). However, because sEVD does not fully decompose the data matrix, the partial factor scores need to be obtained by projecting each individual table onto the same three convex spaces with the PL1L2 algorithm. Formally, the optimization problem that derives $\mathbf{p}_\ell^*$ is represented by the following projection:

$$\mathbf{p}_\ell^* = \text{proj}_{\mathcal{L}_1 \cap \mathcal{L}_2 \cap \mathbf{P}_\perp^*}\left(\mathbf{S}_+ \mathbf{p}_\ell^*\right), \tag{57}$$

with the initial value of $\mathbf{p}_\ell^*$ being the ℓth eigenvector from the plain EVD of $\mathbf{S}_+$ (see, for details, Guillemot et al.[10]). Here, $\mathcal{L}_1$ denotes the $\mathcal{L}_1$-ball, $\mathcal{L}_2$ denotes the $\mathcal{L}_2$-ball, $\mathbf{P}_\perp^*$ denotes the orthogonal space, and together they form a convex set. Consequently, the equation to compute the factor scores (cf. Equation 54) and the partial factor scores (cf. Equation 55) can be rewritten as

$$\mathbf{f}_\ell = \text{proj}_{\mathcal{L}_1 \cap \mathcal{L}_2 \cap \mathbf{P}_\perp^*}\left(\mathbf{S}_+ \mathbf{p}_\ell^*\right)\boldsymbol{\Lambda}^{-\frac{1}{2}}, \tag{58}$$

and

$$\mathbf{f}_{k\ell} = \text{proj}_{\mathcal{L}_1 \cap \mathcal{L}_2 \cap \mathbf{P}_\perp^*}\left(\mathbf{S}_k \mathbf{p}_\ell^*\right)\boldsymbol{\Lambda}^{-\frac{1}{2}}. \tag{59}$$

However, with sparsification, we lose the barycentric property:

$$\sum_{k=1}^{K} \beta_k^2 \mathbf{f}_{k,\ell} \neq \mathbf{f}_\ell. \tag{60}$$

The other way of sparsifying DiSTATIS/CovSTATIS is to sparsify the R_V matrix. Just like in the same approach to sparsify STATIS, we use Algorithm 1 to obtain several sparse and non-negative pseudo-eigenvectors to derive weights and use them to compute separate subspaces of the compromise. To sparsify the EVD of the R_V matrix $\mathbf{C}$ (cf. Equation 31), we use the iterative algorithm of sEVD to solve the same optimization problem as in Equation (42) which searches for the pseudo-eigenvalues ($\hat{\omega}_r$) and the pseudo-eigenvectors ($\mathbf{u}_r$) of $\mathbf{C}$. From the solution, we derive the weights for all K tables (i.e., β_{kr}) from $\mathbf{u}_r$ for the rth subspace:

$$\beta_{k,r}^* = u_{kr}\left(\mathbf{1}^\top \mathbf{u}_r\right)^{-1}. \tag{61}$$

With β_{kr}, we construct the rth subspace of the compromise as

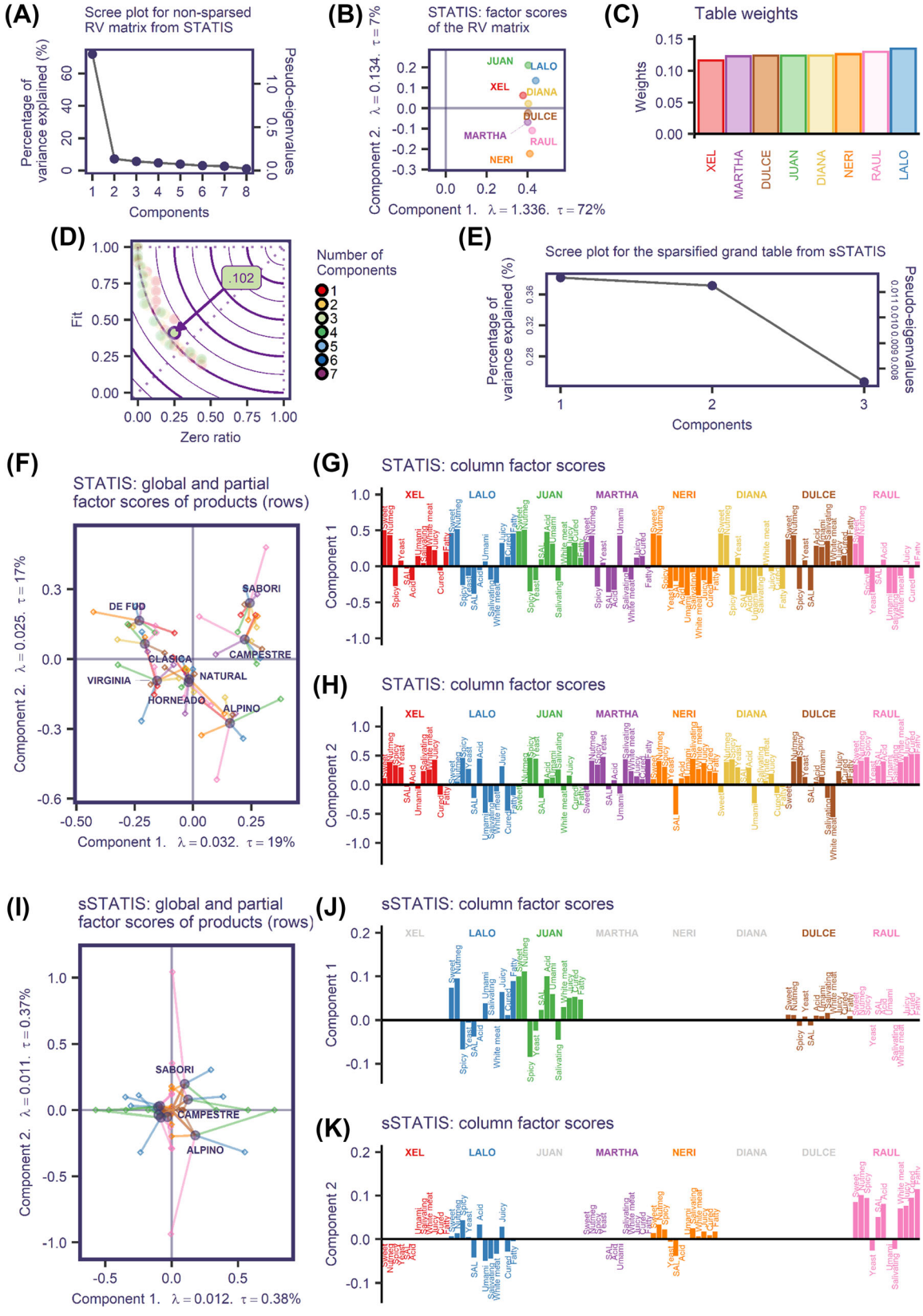

FIGURE 4 Legend on next page.

FIGURE 4 Results from STATIS and sSTATIS when the sparsification is performed with the group constraints. A shows scree plot of the EVD of the R_V matrix with the factor scores shown in B. The scores of the first component (i.e., the horizontal axis) of B are used to derive table weights that are used to build the grand table; the derived weights are illustrated by the barplot in C. D illustrates the sparsity index figure with the optimal sSTATIS solution which identifies 3 components and has a sparsity index of .102. The scree plots of these three sparse components are shown in E. F and I show the scatterplot of the row factor scores respectively from STATIS and from sSTATIS. In these figures, the solid points illustrate the global factor scores that represent the products in the component space, and the hollow diamonds connected to these global factor scores illustrate the partial factor scores which represent how each product is seen from the perspective of the different raters. The two row factor scores plots show a similar pattern that separates the products with sSTATIS illustrating a more compact space. The barplots in G–H and J–K illustrate the column factor scores which represent how the attributes of different raters load onto these components. In these figures, the bars are organized and colored according to the raters. Each bar represents an attribute and, for each block, the bars are presented in the same order: sweet, nutmeg, spicy, yeast, salty, acid, umami, salivating, white meat, juicy, cured, and fatty. In sSTATIS, as shown in G–H, the sparsified tables are grayed out; some non-zero loadings are too small to be seen on the same scale, but the colored variable names are still presented along the axis. The column factor scores from sSTATIS identify Lalo, Juan, Dulce, and Martha for Component 1, and Xel, Lalo, Martha, Neri, and Raul for Component 2. λ denotes the (pseudo-)eigenvalues and τ denotes the proportion of explained variance.

$$\mathbf{S}_{+,r} = \sum_{k=1}^{K} \beta^*_{k,r} \mathbf{S}_k. \tag{62}$$

Next, these R subspace matrices are each decomposed by an EVD:

$$\mathbf{S}_{+,r} = \mathbf{P}^*_r \boldsymbol{\Lambda}_r \mathbf{P}^{*\top}_r \quad \text{such that} \quad \mathbf{P}^{*\top}_r \mathbf{P}^*_r = \mathbf{I}. \tag{63}$$

The global factor scores and the partial factor scores are computed for each of the rth subspaces as

$$\begin{aligned} \mathbf{F}_r &= \mathbf{S}_{+,r} \mathbf{P}^*_r \boldsymbol{\Lambda}_r^{-\frac{1}{2}}, \\ \mathbf{F}_{k,r} &= \mathbf{S}_{k,r} \mathbf{P}^*_r \boldsymbol{\Lambda}_r^{-\frac{1}{2}}. \end{aligned} \tag{64}$$

Because these subspaces are decomposed by the regular EVD, the barycentric property of the partial factor scores remains intact.

4 | ILLUSTRATIONS

We used two sensory evaluation data sets as examples to illustrate these sparse methods. Because DiSTATIS and CovSTATIS only differ in the type of data analyzed and whether they are double-centered, for the purpose of illustration, we only included the example for DiSTATIS.

4.1 | MFA/sMFA and STATIS/sSTATIS

4.1.1 | Data

To illustrate MFA, STATIS, and their sparse methods, we used a sensory evaluation data set in which a trained sensory panel of eight panelists rated the same eight turkey breast products from Mexico on the same 12 attributes. These raters had been trained to identify and measure specific attributes of products. These attributes were rated on a 0 to 9 10-point Likert scale and included sweet, nutmeg, spicy, yeast, salty, acid, umami, salivating, white meat, juicy, cured, and fatty. In this example, each table corresponds to a rater and describes the nine products (on the rows) rated by the 12 attributes (on the columns); rows and the columns are organized in the same order across tables.

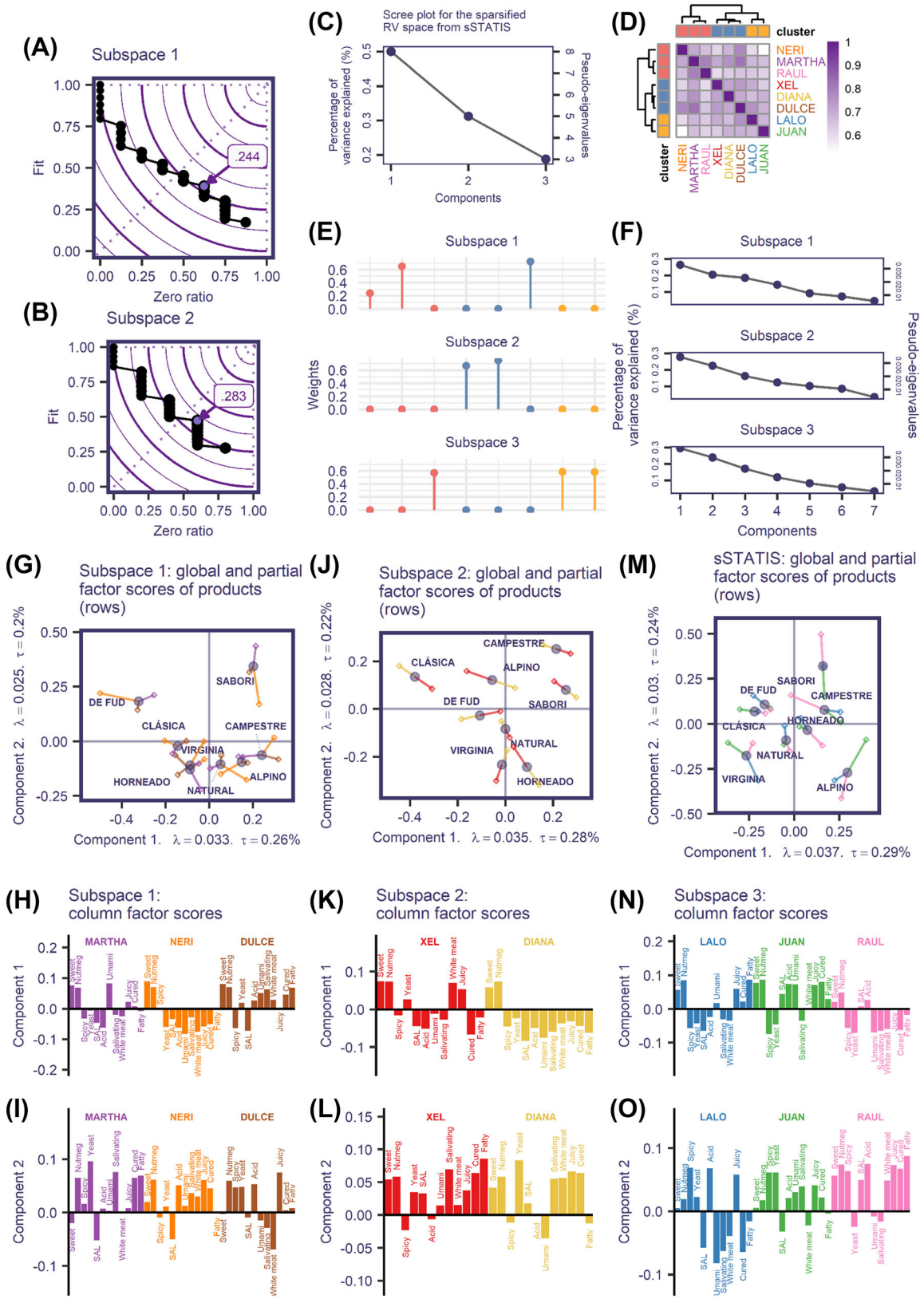

FIGURE 5 Legend on next page.

FIGURE 5 Results from sSTATIS when the R_V matrix is sparsified. The optimal sparse solution to obtain the first subspace is shown in A and the second subspace in B. The variance explained by the three subspaces together give the scree plot in C. The hierarchical clustering analysis of the R_V matrix identifies 3 clusters (shown in D) that are consistent with the tables identified by each subspace as shown in E. F then shows the scree of each of these subspaces and the components are illustrated in G–O. G, J, and M give the row factor scores of the three subspaces and illustrate distinct ways of separating the products. In these figures, the solid points illustrate the global factor scores that represent the products in the component space, and the hollow diamonds connected to these global factor scores identify the partial factor scores that represent how each product is seen from the perspective of different raters. In these figures, the bars are organized and colored according to the raters. Each bar represents an attribute and, for each block, the bars are presented in the same order: sweet, nutmeg, spicy, yeast, salty, acid, umami, salivating, white meat, juicy, cured, and fatty. As shown in the barplots, the first subspace is characterized by Martha, Neri, and Dulce (H–I), the second subspace is characterized by Xel and Diana (K–L), and the third subspace is characterized by Lalo, Juan, and Raul (N–O). λ denotes the (pseudo-)eigenvalues and τ denotes the proportion of explained variance.

4.1.2 | MFA and sMFA results

The results from regular MFA are shown in Figure 3C–E. Figure 3C shows that the first component from MFA explains 25% of the variance and distinguishes Sabori, Campestre, and Alpino from the other products. In general, this separation is associated with the difference between the intense flavors, such as spicy and salty, and the more pleasant flavors, such as sweet and nutmeg (see Figure 3D). This component also identifies two patterns among the raters: Raul, Diana, and Neri versus the rest. The second component from MFA explains 20% of the variance and distinguishes Sabori, Campestre, de Fud, and Classica from the other products (see Figure 3C). Figure 3E shows that this separation corresponds to difference between the salt-related tastes, such as cured and umami, versus smell and trigeminal attributes.

With sparsification, the extracted components seek to concurrently (1) maximize the proportion of variance explained and (2) keep only the most representative tables. We identify the optimal sparsification solution that has more than 1 dimension and the largest sparsity index. The optimal solution of sMFA is a two-component solution (see Figure 3A for the scree plot) identified by a sparsity index of .112 (see Figure 3B). Each of these two components explains around 1.5% of the variance. In sMFA, we identify similar separations between the products (see Figure 3F) and the most informative corresponding tables for Components 1 and 2. For Component 1, we identify four raters (i.e., Lalo, Juan, Martha, and Dulce) who have a consistent pattern of differentiating the intense flavors from the pleasant ones (Figure 3G). For Component 2, we identify five raters (i.e., Lalo, Juan, Martha, Dulce, and Raul) who in general distinguish the salt-related tastes from smell and trigeminal attributes.

4.1.3 | STATIS and sSTATIS results

The same data set was also analyzed with STATIS and sSTATIS. The results from STATIS and its two ways of sparsification are shown respectively in Figures 4 and 5. Figure 4 shows the results from STATIS and sSTATIS implemented by the sSVD with group constraints. Figure 4A–C shows the EVD results of the R_V matrix. The scree plot in Figure 4A showed that the R_V matrix has a strong first component which captures the general commonality of the tables (i.e., the raters) as shown also in the first component (i.e., the horizontal axis) of Figure 4B. This first component is then used to obtain the weights (shown in Figure 4C) that are applied to build the grand table. The final results from STATIS give a similar pattern to those from MFA. As shown in Figure 4F–H, the first STATIS component explains 19% of the variance of the grand table and distinguishes Sabori, Campestre, and Alpino from the other products (along the horizontal axis in Figure 4F). As shown in Figure 4G, this separation is associated with the difference between the intense, spicy flavors, such as spicy and salty, and the more pleasant flavors, such as sweet and nutmeg. The second STATIS component explains 17% of the variance of the grand table and distinguishes Sabori, Campestre, de Fud, and Classica from the other products (along the vertical axis in Figure 4F). Again, similar to MFA, this separation is associated with the difference between the salt-related tastes versus smell and trigeminal attributes.

The optimal non-unidimensional sparsification results of sSTATIS with the modified sSVD is a three-component solution with a sparsity index of 0.102 (see Figure 4D). Figure 4E shows the scree plot with pseudo-eigenvalues of these three components. The first two components of sSTATIS explain respectively 0.38% and 0.37% of the variance of the grand table. The global factor scores also show a similar, but much smaller, separation as in sMFA (see Figure 4I). The

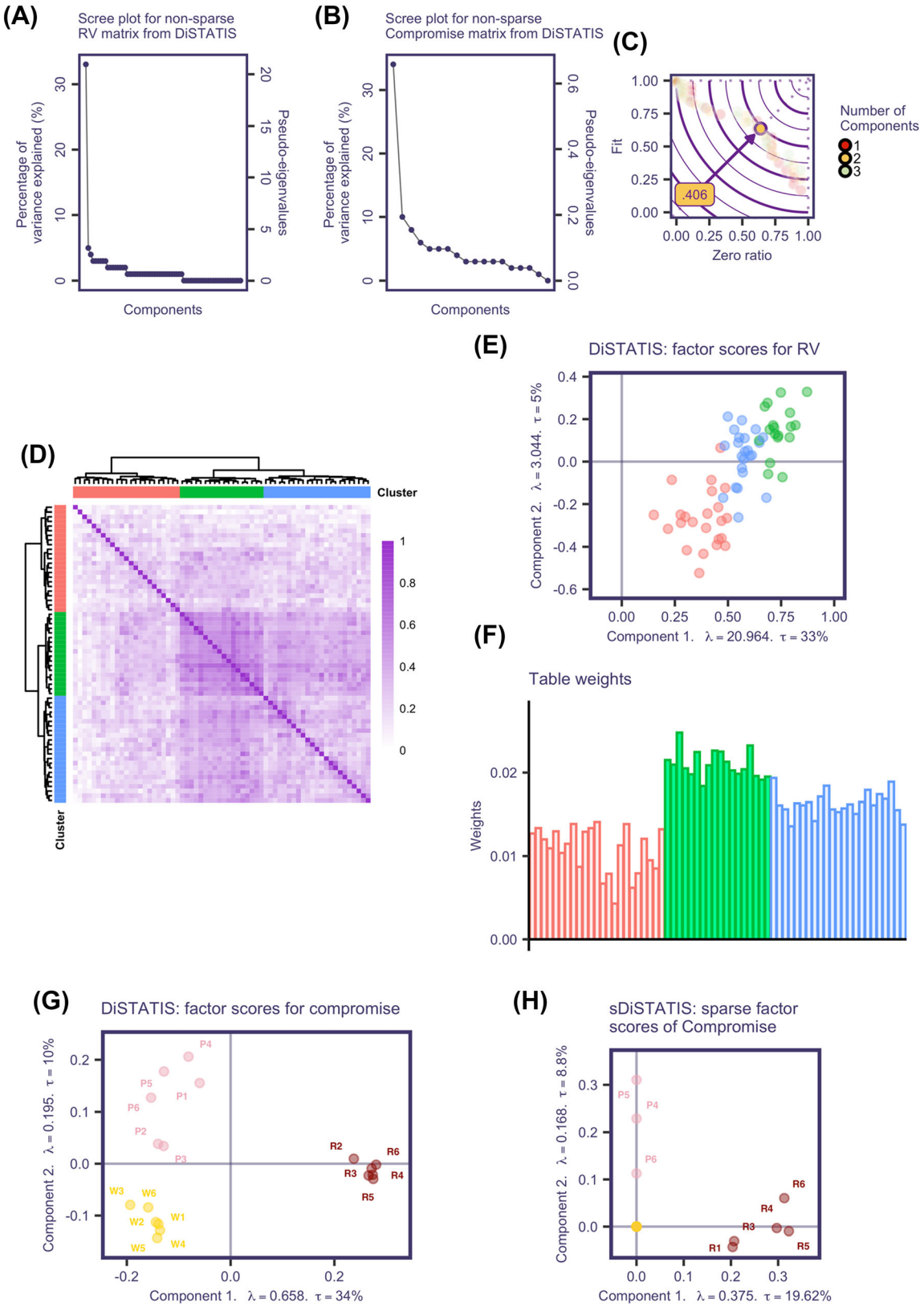

FIGURE 6 Legend on next page.

FIGURE 6 Results from regular DiSTATIS, and sDiSTATIS when the compromise matrix is sparsified. The R_V matrix is represented as a heatmap in D, along with the three clusters of assessors identified with hierarchical clustering analysis. A and B show respectively the scree plots of the R_V and the compromise. The factor maps for the R_V and the compromise are shown respectively in E and G, and the weights used to compute the compromise are shown in F. The sparse decomposition of the compromise is shown in H, using the optimal sparse parameter identified by the sparsity index plot in C. λ denotes the (pseudo-)eigenvalues and τ denotes the proportion of explained variance.

largest separation between products is the one between Sabori and Alpino on the second component. The partial factor scores that are illustrated by the colored lines extended from each global factor score further show how the tables are sparsified. Figure 4J,K shows a more direct view of how the tables are sparsified from the column factor scores. The first component (Figure 4J) identifies four raters (i.e., Lalo, Juan, and Dulce) who have a consistent pattern of differentiating the intense flavors from the pleasant ones. The second component (Figure 4K) identifies three raters (i.e., Lalo, Neri, and Raul) who distinguish the salt-related tastes from smell and trigeminal attributes. These three raters are also the ones who are weighted the most in the grand table (see Figure 4C).

The results of the implementation of sSTATIS that sparsifies the R_V matrix are shown in Figure 5. Figure 5A,B shows the optimal non-unidimensional solutions for the first two subspaces, and the third subspace which is the residual of the first two. Figure 5C shows the scree plot of pseudo-eigenvalues of the R_V matrix. Figure 5D shows the R_V matrix between tables with the 3 clusters from hierarchical clustering analysis, and Figure 5E shows that the identified tables (colored by their clusters) which construct these subspaces are partitioned in a way similar to the clusters. For the three subspaces, Figure 5F–O gives the EVD results including the scree plot (Figure 5F), the global and partial row factor scores (Figure 5G–I), and the columns vector scores (Figure 5J–O). From the row factor scores, sSTATIS identifies three patterns of product separation. For example, the first component of Subspace 1 distinguishes Sabori, Campestre, and Alpino from de Fud, whereas the first component of Subspace 2 is driven by the difference between Sabori and Campestre versus Classica, and that of Subspace 3 distinguishes Sabori, Campestre, and Alpino from Classica and Virginia. The consistent products that contributes to the first components of these subspaces are Sabori and Campestre, which also correspond to ratings in sweet and nutmeg as shown in Figure 5J–L. The first components of different subspaces differentiate these two products from de Fud in Subspace 1, from Classica in Subspace 2, and from Classica and Virginia in Subspace 3. The second components of these subspaces are distinguishing different sets of products based on different featuring flavors. For example, the second component of Subspace 1 is driven by ratings of sweet and salt-related tastes, the second component in Subspace 2 is driven by ratings of spicy and acidity, and the second component in Subspace 3 is driven by the difference between the salt-related tastes and all other attributes.

4.2 | DiSTATIS and sDiSTATIS

4.2.1 | Data

For DiSTATIS/sDiSTATIS, we analyzed a different data set from a free sorting task (original data from Ballester et al.[19]). In this example, 64 assessors sorted, only by smell, 18 wines which can be further characterized as six red, six white, or six rosé wines. The sorting result of each assessor gives one distance matrix. When two wines are sorted together, their distance is 0; when two wines are sorted in two different groups, their distance is 1.

4.2.2 | Results

The results from DiSTATIS and its two ways of sparsification are shown respectively in Figures 6 and 7. Figure 6 shows the results from DiSTATIS and from sDiSTATIS when only the compromise matrix is sparsified. Figure 7 shows the results from sDiSTATIS where only the R_V matrix is sparsified.

The scree plot in Figure 6A shows that the R_V matrix has a strong first dimension, which captures the general commonality of the dissimilarity tables (i.e., the assessors) as shown also in the first component (i.e., the horizontal axis) of Figure 6E. The R_V matrix itself is represented on Figure 6E as a heatmap on which a hierarchical clustering was

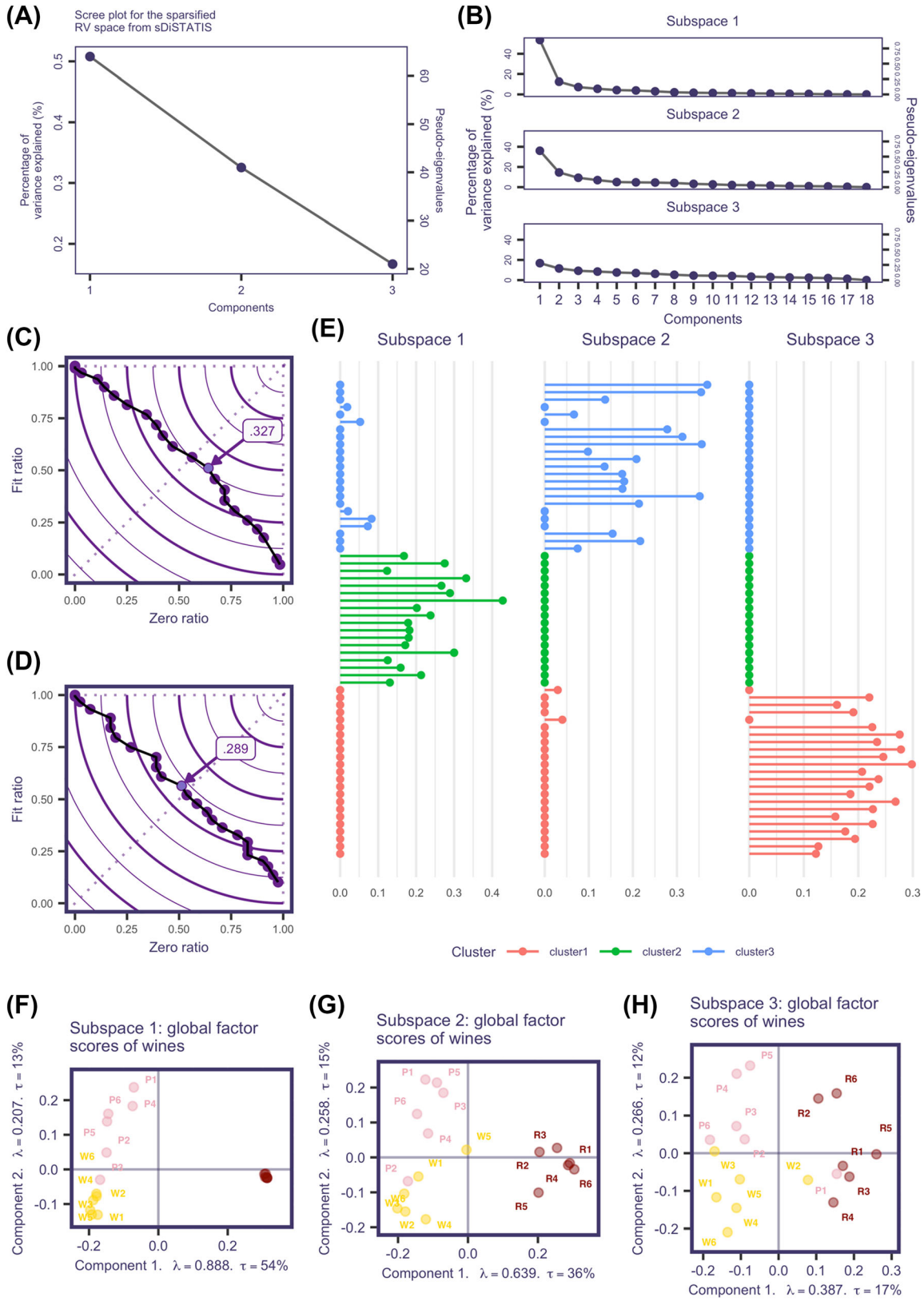

FIGURE 7 Legend on next page.

FIGURE 7 Results from DisSTATIS when the the R_V matrix is sparsified. The optimal sparse solution to obtain the first subspace is shown in C and the second subspace in D. The variance explained by the three subspaces together give the scree plot in A. The hierarchical clustering analysis of the R_V matrix identifies 3 clusters that are consistent with the tables identified by each subspace as shown in E. B then illustrates the scree of each of these subspaces, and their components are illustrated in F–H. F, G, and H give the global factor scores of the three subspaces and illustrate distinct ways of separating the wines. λ denotes the (pseudo-)eigenvalues a nd τ denotes the proportion of explained variance.

performed, identifying three groups of assessors that are represented with the same color scheme in Figures 6 and 7. The first component of the R_V matrix is then used to obtain the weights (shown in Figure 6F) that are applied for the individual tables to build the compromise matrix. The scree plot in Figure 6B shows that the resulting compromise matrix also has a strong first component. The final results from the decomposition of the compromise matrix are shown in Figure 6G for the regular decomposition (on the left), and in Figure 6H for the sparse decomposition (on the right). The first components of the regular and the sparse analyses explain, respectively, 34% and 19% of the variance and distinguish the red wines from the others (along the horizontal axis in Figure 6G,H). The second components of the regular and the sparse analyses explain, respectively, 10% and 9% of the variance and distinguish white from rosé wines (along the vertical axes in Figure 6G,H). Figure 6C displays the sparsity index plot that is used to select the optimal sparsity parameter (associated with a value of .406 for the sparsity index) for the sparse decomposition of the compromise.

The results obtained by sparsifying the R_V matrix are shown in Figure 7. Figure 7C,D shows the optimal solutions for the first two subspaces, and the third subspace is the residual of the first two. Figure 7A shows the scree plot of pseudo-eigenvalues from all three subspaces. Figure 7 shows that the identified tables (colored by their clusters) that construct these subspaces are close to the identified clusters from hierarchical clustering (shown in Figure 6E). Figure 7F–H illustrates the global factor scores associated with the three subspaces. From the row factor scores, sDiSTATIS identifies three patterns of how the assessors distinguish the wines. Based on the factor maps of their respective compromise matrices, it appears that the different subspaces distinguish between assessors of decreasing ability to distinguish between wines.

4.3 | Conclusion and perspectives

Analyzing data with a strong structure is challenging: Statisticians and practitioners need new methods to help them explore the data while keeping the structure (in blocks, or sub-tables) easy to interpret. To tackle this problem, we developed sparse extensions of three multi-table data analysis techniques: MFA, STATIS, and DiSTATIS/covSTATIS. These new sparse methods are based on integrating sparse and group-sparse constraints with the SVD and the EVD.

We showed that the ability to include group sparsity constraints allows us to sparsify variables in groups, therefore as tables, which is especially useful for sMFA and sSTATIS. Beyond sparsification, we included orthogonality constraints while sparsifying the components to preserve their ease of interpretation. Furthermore, for STATIS methods, the algorithm that we proposed searches for several pseudo-eigenvectors of the R_V matrix that are sparse, orthogonal, and non-negative. Such a decomposition is achieved by successively applying a sEVD to the sub-matrix (of the R_V matrix) that is composed of tables that have strictly positive weights derived from the previously estimated pseudo-eigenvectors.

We applied sMFA and sSTATIS to a sensory evaluation data set of ratings on turkey breast by a panel. The results from both analyses showed that, although there were slight differences given different weighting strategies, sMFA and sSTATIS still behaved very similarly likely because of the homogeneity of the sub-tables in this example. However, the results could be more different when the data tables are more heterogeneous. In such cases, users will need to choose between the two methods by deciding whether the average table structure (from sMFA) or the most common table structure (from sSTATIS) best represents the data set. However, when analyzing data tables with high heterogeneity, our results suggest that the version of sSTATIS which sparsifies the R_V matrix is the best approach. This approach identifies different patterns of data structures by extracting subspaces from the R_V space; therefore, it can stratify clusters of patterns to analyze the individual differences among the data tables. As for sDiSTATIS, because the data tables are symmetric, we used another data set of a free sorting task to illustrate the method. Similar to STATIS, we showed that the "subspace" algorithm identified homogeneous groups of tables. We also found that these groups were comparable to

the ones obtained by hierarchical clustering clustering. Finally, when we sparsify the R_V, we need to specify the number of subspaces. Results from hierarchical clustering analysis is a good start but other clustering methods could be applied to decide how many subspaces could be explored. Here in our wine tasting example, we identified three subspaces and these subspaces were different in how distinct the assessors thought the wines were.

In this paper, the other sDiSTATIS approach where the compromise is sparsified clearly differ from sMFA and sSTATIS. This sDiSTATIS approach is the only method, of all those that we mentioned, that sparsified the products. This effect facilitates the interpretation of components by extracting a more *simple* component structure and is useful when the data measure a large set of products. In this sDiSTATIS, the steps that consider possible individual differences among the data tables are the exact same steps as in regular DiSTATIS: computing the R_V and deriving the weights for the tables. Therefore, different from other sparse multi-block methods in this paper, the sparsification algorithm *per se* of this approach of sDiSTATIS does not take individual differences into account.

It is worth mentioning that, with sparsification, the barycentric property of partial factor scores was kept for sMFA, sSTATIS, and the sDiSTATIS/sCovSTATIS that sparsifies the R_V matrix. However, this property was lost for sDiSTATIS and sCovSTATIS when the compromise is sparsified. In sMFA and sSTATIS, the pseudo-right singular vectors are directly associated with different tables. When these vectors are sparsified, the tables are naturally sparsified and the partial factor scores are computed directly from these vectors. In sDISTATIS/sCovSTATIS that sparsifies the R_V matrix, because the factor scores are computed from the regular EVD on each subspace, the barycentric property is kept. However, when we sparsify DiSTATIS and CovSTATIS by sparsifying the compromise matrix, we perform an sEVD on it. Thus, the elements in pseudo-eigenvectors correspond to products (i.e., rows and columns of the compromise) instead of assessors (i.e., tables). The partial factor scores of each table need to be obtained by projecting each table as an out-of-sample data set. Because the projecting algorithm is non-linear with sparsification, linear properties such as the barycentric property, are not guaranteed. Such outcome could be due to the feature of the algorithm itself or of the specific example that we used, and more research is needed to investigate when and why the barycentric property could be lost.

Although not demonstrated, it is also computationally possible to (1) sparsify the products (i.e., rows) when sparsifying the tables and to (2) combine the sparsifications of the R_V matrix and of the compromise for STATIS-like methods. In this paper, we did not sparsify the products because of two reasons. First, there are only eight products in our example that give a grand table. Second, we want to focus on how sparsifying a multi-block method can help analyze individual differences among tables. As for combining the two sparsifications for STATIS-like methods, the results from our examples often give components that are too sparse due to their rather simple structures. We argue that such combination only makes sense when the data involve a big group of assessors on a large set of products.

Future developments will focus on developing new constraints, including, but not limited to, group sparsity constraints or network-based constraints. Additional work will also be needed to help statisticians identify optimal values for the parameters associated to these constraints. Furthermore, future applications could be used to compare between sMFA that sparsifies both rows and data tables, and sSTATIS that sparsifies both the R_V and the compromise. More types of data could be analyzed with the methods that we presented: undirected graphs (gene connectivity networks, brain connectivity, phylogenetic trees), similarity based on cross-products (brain connectivity, brain imaging), etc. Overall, we showed that sMFA, sSTATIS and sDiSTATIS facilitate the interpretation of the structure of the data: they help identify homogeneous groups of subjects, groups of tables, and groups of variables.

PEER REVIEW

The peer review history for this article is available at https://publons.com/publon/10.1002/cem.3443.

DATA AVAILABILITY STATEMENT

All programs and data are available at https://github.com/juchiyu/sDiSTATISpaper.

ORCID

Ju-Chi Yu https://orcid.org/0000-0002-6360-1861
Carlos Gómez-Corona https://orcid.org/0000-0003-2928-7597
Vincent Guillemot https://orcid.org/0000-0002-7421-0655

REFERENCES

1. Pagès J. Eléments de comparaison entre l'analyse factorielle multiple et la méthode statis. *Rev Stat Appl*. 1996;44:81-95.
2. Abdi H, Williams LJ, Valentin D, Bennani-Dosse M. STATIS and DISTATIS: optimum multitable principal component analysis and three way metric multidimensional scaling. *Wiley Interdiscip Rev: Comput Stat*. 2012;4(2):124-167.
3. Thurstone LL. *The Vectors of Mind: Multiple-Factor Analysis for the Isolation of Primary Traits*. University of Chicago Press; 1935.
4. Thurstone LL. Multiple-factor analysis; a development and expansion of the vectors of mind; 1947.
5. Kaiser HF. The varimax criterion for analytic rotation in factor analysis. *Psychometrika*. 1958;23(3):187-200.
6. Cattell RB. *The Scientific Use of Factor Analysis in Behavioral and Life Sciences*. Plenum; 1978.
7. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media; 2009.
8. Journée M, Nesterov Y, Richtárik P, Sepulchre R. Generalized power method for sparse principal component analysis. *J Mach Learn Res*. 2010;11(2):517-553.
9. Witten DM, Tibshirani R, Hastie T. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*. 2009;10(3):515-534.
10. Guillemot V, Beaton D, Gloaguen A, et al. A constrained singular value decomposition method that integrates sparsity and orthogonality. *PLOS ONE*. 2019;14(3):e0211463.
11. Trendafilov NT, Adachi K. Sparse versus simple structure loadings. *Psychometrika*. 2015;80(3):776-790.
12. Gloaguen A, Guillemot V, Tenenhaus A. An efficient algorithm to satisfy L1 and L2 constraints. In: 49èmes Journées de Statistique; 2017; Avignon, France.
13. Liu R, Niang N, Saporta G, Wang H. Sparse correspondence analysis for contingency tables; Advances in Data Analysis and Classifications; ; 2023.
14. Trendafilov NT, Fontanella S, Adachi K. Sparse exploratory factor analysis. *Psychometrika*. 2017;82(3):778-794.
15. Abdi H, Williams LJ, Valentin D. Multiple factor analysis: principal component analysis for multitable and multiblock data sets. *Wiley Interdiscip Rev: Comput Stat*. 2013;5(2):149-179.
16. Pagès J. *Multiple Factor Analysis by Example Using R*. CRC-Press; 2015.
17. van den Berg E, Schmidt M, Friedlander MP, Murphy K. Group sparsity via linear-time projection, Department of Computer Science, University of British Columbia Technical Report; 2008.
18. Robert P, Escoufier Y. A unifying tool for linear multivariate statistical methods: the RV-coefficient. *Appl Stat*. 1976;25:257-265.
19. Ballester J, Abdi H, Langlois J, Peyron D, Valentin D. The odor of colors: can wine experts and novices distinguish the odors of white, red, and rosé wines? *Chemosens Percept*. 2009;2:203-213.

How to cite this article: Yu J-C, Gómez-Corona C, Abdi H, Guillemot V. Sparse Multiple Factor Analysis, sparse STATIS, and sparse DiSTATIS with applications to sensory evaluation. *Journal of Chemometrics*. 2023; e3443. doi:10.1002/cem.3443

APPENDIX A: NON-NEGATIVITY AND ORTHOGONALITY CONSTRAINTS

Lemma 1. Let $(\mathbf{a},\mathbf{b}) \in \mathbb{R}^K$ $(K > 2)$. If $\mathbf{a}$ and $\mathbf{b}$ are orthogonal and non-negative, then, for all $k = 1,\ldots,K$

$$a_k \neq 0 \Rightarrow b_k = 0.$$

Proof. Since $\mathbf{a}$ and $\mathbf{b}$ are non-negative vectors, then for their cross-product to be null, all elements of the cross-product need to be null. Therefore, for each element $a_k b_k$ of the cross-product needs to be null, which means at least one of a_k or b_k needs to be null. In other words, since $\mathbf{a}$ and $\mathbf{b}$ are non-negative vectors, then

$$\sum_{k=1}^{K} a_k b_k = 0 \Leftrightarrow a_k b_k = 0, \forall k = \{1, \ldots, K\},$$

which completes the proof. □