CoVariance Neural Networks

Principal Component Analysis Meets Learning with Graphs

Saurabh Sihag, Andrea Cavallo, Elvin Isufi, Gonzalo Mateos, and Alejandro Ribeiro†

INTRODUCTION

Covariance matrices encode linear (statistical) dependencies between different pairs of features within a dataset and have been the cornerstone of multivariate data analysis in a host of applications characterized by spatially distributed data acquisition protocols. Noteworthy examples with signal processing (SP) and machine learning (ML) relevance include neuroimaging [1], computer vision [2], weather modeling [3], traffic flow analysis [4], and cloud computing [5], to name a few. Principal Component Analysis (PCA) is among the most widely adopted covariance matrix-driven statistical techniques [6]. Specifically, PCA is an orthogonal linear transformation derived from the eigenvectors of the covariance matrix [7]. Because the eigenvectors and their corresponding eigenvalues of the covariance matrix are individually tied to the variance explained in a given dataset, PCA is often used as a dimensionality reduction technique that enjoys well-documented optimality properties. In a nutshell, one projects a dataset onto a lower-dimensional covariance eigenspace with the goal of preserving the "most signal" for a given learning task. A schematic illustration of a prototypical PCA-driven regression model is provided in Fig. 1 and Fig. 3 (left) and consists of two major modules: (a) PCA transform on the input data that can be viewed as a (dimensionality-reducing) feature extractor; followed by (b) a linear regression model that assigns "importance" weights to individual principal components.

In practice, all one has to work with are empirical covariance matrices estimated from data (referred to as sample covariance matrices), which are statistical estimates of the population covariance matrix. In this

†Saurabh Sihag is with the Department of Electrical and Computer Engineering at the University at Albany, SUNY, Albany, NY (email: ssihag@albany.edu). Andrea Cavallo (email: A.Cavallo@tudelft.nl) and Elvin Isufi (email: e.isufi-1@tudelft.nl) are with the Delft University of Technology, Delft, The Netherlands. Gonzalo Mateos is with the Department of Electrical and Computer Engineering at the University of Rochester, Rochester, NY (email:gmateosb@ece.rochester.edu). Alejandro Ribeiro is with the Department of Electrical and Systems Engineering at the University of Pennsylvania, Philadelphia, PA (email:aribeiro@seas.upenn.edu).

context, a first key challenge facing PCA-based learning approaches is the lack of reproducibility of findings [8], [9], which stem from finite sample-induced stochastic perturbations in the eigenvector estimates associated with *close* eigenvalues of the sample covariance matrix [10]; see also 'Challenges to Learning with PCA' and Fig. 1. Moreover, conventional PCA-driven models are not adept at multiscale information processing needed for applications where data are acquired at different e.g., spatial scales; such as in geosciences, atmospheric sciences, power grids, and brain imaging [11]. This is because as the number of input features changes, so does the size of the covariance matrix and the properties of its eigenspace – which needs to be recomputed from scratch, leading also to computational issues. Hence, fundamental theoretical limits of inference across multiscale datasets (while effectively leveraging their redundancies) are categorically lacking. Furthermore, the PCA-based feature preprocessing step is decoupled from the downstream learning task. In 'GSP-Driven Implementation of PCA', we pivot towards a graph-driven perspective to covariance matrix, which ties PCA to the domain of graph signal processing [12]. This novel perspective provides an alternative, linear-shift-and-sum implementation in terms of the covariance matrix for PCA-driven learning models, while addressing the aforementioned challenges to its practical implementation and laying the groundwork for significant conceptual advancements to PCA.

Leveraging covariance matrices as graph representations (for example, covariance matrices derived from functional brain measurements in network neuroscience; see Fig. 2) is ubiquitous in graph-driven machine learning frameworks. For instance, a graph neural network (GNN) can operationally exploit the graph structure manifested in the form of covariance matrices for learning. Recent advances in graph signal processing (GSP) have provided key operational and theoretical principles for GNNs [13]–[16]. We begin the theoretical discussions by elucidating the natural connection between PCA and theoretical understanding of the fundamental computational block within GSP: a graph convolutional filter that has a linear-shift-and-sum implementation in the matrix representation of a graph. Spectral analysis of the graph filter reveals that its outcome is tied to data-driven exploitation of the eigenspectrum of the graph [17]. Setting the graph to be a covariance matrix within a graph filter reveals an implementation of a learning model that is conceptually identical to PCA (Fig. 3). Building upon the said equivalence between PCA-based learning and graph filters over covariance matrices, recent works have undertaken the mathematical study of GNNs with covariance matrices as graphs, termed *coVariance neural networks (VNNs)* [18]–[21]. A key aim of this tutorial is to convey the broad scope of improvements and generalization capabilities offered by VNN models over PCA-based and other related learning pipelines. In this context, our aim is not to diminish the PCA transform or put VNNs in contrast to it, but rather to offer significant conceptual advancements to PCA through VNN models.

In 'Stability of VNNs', we discuss the theoretical results underlying the stability of VNN outcomes

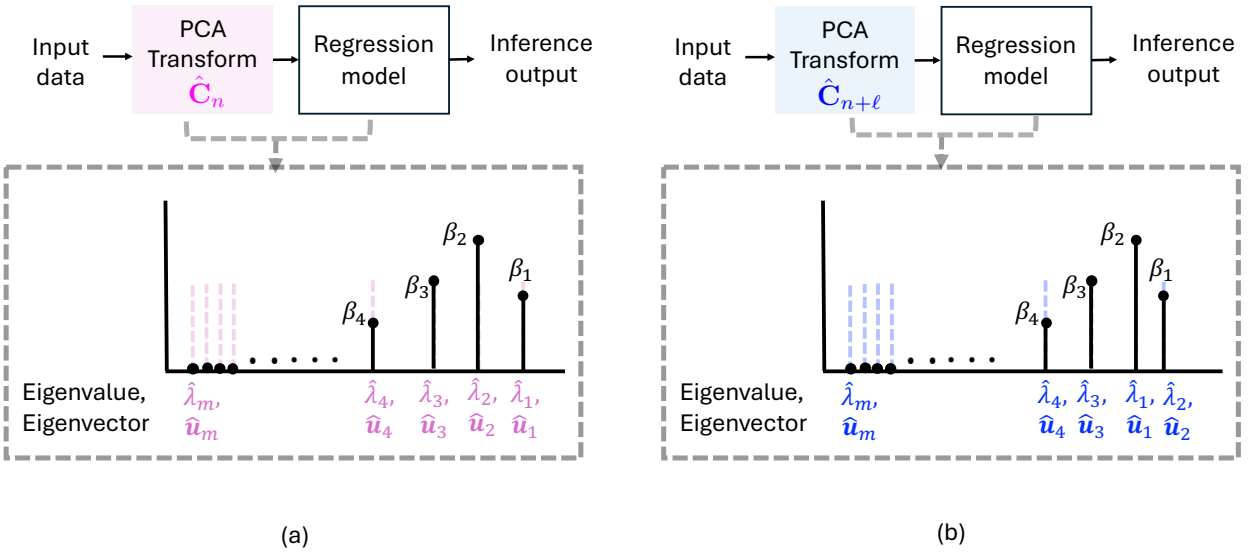

Fig. 1. Potential instability of PCA-driven regression model illustrated through an example. (a) PCA-driven regression model consists of implementing a PCA transform based on $\hat{\mathbf{C}}_n$ first, followed by a linear regression model. The learned coefficients $\{\beta_1, \ldots, \beta_m\}$ of the linear model are determined from a given training dataset and represent the data-driven importance of individual eigenvectors of $\hat{\mathbf{C}}_n$ to the learning objective. (b) Consider the scenario where the sample covariance matrix is re-estimated with $n+\ell$ samples to yield $\hat{\mathbf{C}}_{n+\ell}$ and the learned coefficients $\{\beta_1, \ldots, \beta_m\}$ are inherited from (a). If the estimate $\hat{\lambda}_1$ of first eigenvalue of $\mathbf{C}$ in $\hat{\mathbf{C}}_{n+\ell}$ is smaller than that of its estimate $\hat{\lambda}_2$ of the second eigenvalue of $\mathbf{C}$ due to the finite sample effect, the coefficients β_1 and β_2 are associated with the estimates of eigenvectors $\mathbf{u}_2$ and $\mathbf{u}_1$, respectively. Consequently, the higher importance (in terms of β_1) placed on $\mathbf{u}_2$ relative to $\mathbf{u}_1$ in setting (b) is in contrast to (a). This may lead to potentially inconsistent regression performance relative to (a) as the eigenvectors of a covariance matrix are orthogonal to each other.

to stochastic perturbations in the sample covariance matrix. As an upshot of the theoretical analyses on the stability of VNNs, we also elucidate conditions under which VNNs offer guaranteed reproducibility of learning outcomes on independently collected datasets, leading to an inherently stable deep learning alternative to PCA-based pipelines. While existing studies on GNNs with graph convolutional filters have established stability to abstract (absolute or relative) perturbations [15], the *stability results for VNNs* discussed herein are markedly refined by borrowing precise error models and mathematical tools from perturbation theory of covariance matrices [10].

In 'Transferability of VNNs', we leverage the mathematical lens of graphons and limit objects of graph sequences to demonstrate that VNNs exhibit transference across multiscale datasets (datasets capturing the same information with distinct numbers of features or scales). Rigorous analysis of the transferability of VNNs across multiscale datasets is key to understanding how the redundancy of information within covariance matrices of different dimensionalities could be fruitfully exploited for principled

design and efficient deployment of deep NN models in communications, array processing, and neuroimaging analysis systems.

Recent advances in graph signal processing (GSP) and spectral theory of GNNs have provided key operational and theoretical principles for stability or transferability of GNNs [13]–[16]. However, there are critical theoretical gaps between the nuances of spatial and spectral information inherent to a covariance matrix and the known mathematical and empirical characteristics of GNNs [14], thus, inhibiting *principled* design and application of GNNs across broad domains where covariance matrices emerge as the go-to descriptor of multivariate data structure. This article lays the groundwork for rigorous theoretical studies of GNNs with covariance matrices as graphs, which have the potential to elucidate a principled understanding underlying the deployment of deep neural networks (NNs) across a wide spectrum of timely SP applications. All in all, the surveyed theoretical analyses of VNN stability and transferability properties are rooted at the crossroads of statistical SP and (geometric) deep learning, offering an ideal framework to bridge the gap between traditional statistical approaches using covariance matrices and the recently developed insights and capabilities offered by GNNs.

LEARNING WITH COVARIANCE MATRICES: A PCA PERSPECTIVE

Covariance matrices encode pairwise, linear statistical dependencies in a given dataset. Formally, for a dataset consisting of n random, independent, and identically distributed (i.i.d) data samples $\mathbf{x}_i \in \mathbb{R}^m, \forall i \in \{1, \ldots, n\}$. Using the samples $\mathbf{x}_i, \forall i \in \{1, \ldots, n\}$, the empirical covariance matrix is

$$\hat{\mathbf{C}}_n \triangleq \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^{\mathsf{T}} , \tag{1}$$

The sample covariance matrix is a statistical estimate of the true covariance matrix (also referred to as ensemble covariance matrix) $\mathbf{C}$. This ensemble covariance matrix $\mathbf{C}$ is determined from an $m-$dimensional random vector $\mathbf{x} \in \mathbb{R}^m$ as $\mathbf{C} \triangleq \mathbb{E}[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{x} - \mathbb{E}[\mathbf{x}])^{\mathsf{T}}]$. However, the data model for $\mathbf{x}$ or its statistics are often not available in practical applications. The convergence between $\hat{\mathbf{C}}_n$ and $\mathbf{C}$ is stochastic, with its principles governed by perturbation theory [10]. For conciseness, we reuse the notation $\mathbf{x}$ to denote a data sample from the given dataset in subsequent sections of this article.

Principal Component Analysis

PCA leverages the covariance matrix to reveal the hidden, simplified structure of complex data [7]. PCA has been used abundantly across many domains where multivariate datasets appear [6], [9]. The

workhorse PCA is deployed as a linear transformation over the data samples, where the transformation is characterized by the eigenspectrum of the covariance matrix. Given the eigendecomposition

$$\hat{\mathbf{C}}_n = \hat{\mathbf{U}}\hat{\mathbf{\Lambda}}\hat{\mathbf{U}}^{\mathsf{T}} \, . \tag{2}$$

where $\hat{\mathbf{U}} = [\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_m]$ is the orthonormal matrix of m eigenvectors, and $\hat{\mathbf{\Lambda}} = \mathsf{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_m)$ is the diagonal matrix of eigenvalues ordered as $\hat{\lambda}_1 \geq \hat{\lambda}_2 \cdots \geq \hat{\lambda}_m$, the PCA transformation of a data sample $\mathbf{x}$ is given by

$$\tilde{\mathbf{x}} = \hat{\mathbf{U}}^{\mathsf{T}}\mathbf{x} \, . \tag{3}$$

Thus, PCA is an orthogonal, linear transformation that leads to a change of basis of $\mathbf{x}$, with the new basis determined by the eigenvectors of $\hat{\mathbf{C}}_n$. The eigenvectors of $\hat{\mathbf{C}}_n$ are typically referred to as the *principal components*. Because $\hat{\mathbf{C}}_n$ is the estimate of $\mathbf{C}$, its eigenvectors $\hat{\mathbf{U}}$ and eigenvalues $\hat{\mathbf{\Lambda}}$ are estimates of the eigenvectors $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$ and eigenvalues $\mathbf{\Lambda} = \mathsf{diag}(\lambda_1, \ldots, \lambda_m)$, respectively. In practice, we observe $\hat{\mathbf{C}}_n$ and not $\mathbf{C}$ and therefore, the PCA transform is defined in terms of $\hat{\mathbf{U}}$. Furthermore, the eigenvalues of $\hat{\mathbf{C}}_n$ encode the *variance* of the dataset in the specific direction of the corresponding principal component. Specifically, it can be readily shown that [7]

$$\sum_{i=1}^{n} |\hat{\mathbf{u}}_j^{\mathsf{T}}\mathbf{x}_i|^2 = \hat{\lambda}_j \tag{4}$$

Thus, the variance of a dataset along the direction of a specific principal component is commensurate with the magnitude of its corresponding eigenvalue. Equivalently, it could be stated that the principal components *explain* the variance within a given dataset in proportion to the magnitude of their corresponding eigenvalues.

PCA helps elucidate the hidden structure or patterns of interest in the raw data, which are then leveraged for further statistical analyses (such as, clustering, regression, etc.). For instance, the framework for a PCA-driven regression model is illustrated in Fig.3. PCA-driven approaches form the workhorse techniques for statistical analysis across a broad range of learning tasks and data modalities.

Challenges to Learning with PCA

Despite its widespread use, PCA is constrained by several significant limitations regarding reproducibility of inferred outcomes and computational bottlenecks in implementation. The challenges concerning the reproducibility of inferred outcomes arise from the fact that estimating principal components from finite samples is highly sensitive to noise and statistical uncertainty, particularly in low-data regimes or when covariance eigenvalues are closely spaced [8]. According to the Davis-Kahan theorem [22], the estimation error for the first k principal components for a sample covariance matrix is inversely proportional to the

eigengap $\lambda_k - \lambda_{k+1}$. Formally, for any two eigenvectors $\hat{\mathbf{u}}_j$ and $\mathbf{u}_i$ of $\hat{\mathbf{C}}_n$ and $\mathbf{C}$, respectively, their inner product scales with the number of samples n and the eigengap $|\lambda_i - \lambda_j|$ as [10, Theorem 4.1]

$$\mathbb{P}\left(|\langle \hat{\mathbf{u}}_i, \mathbf{u}_j \rangle| \geq t\right) = \mathcal{O}\left(\frac{1}{nt^2(\lambda_i - \lambda_j)^2}\right) , \tag{5}$$

for some constant $t > 0$. The relationship in (5) implies that as eigenvalues become less distinct, the corresponding eigenspaces are harder to separate, leading to inaccurate component estimation. Prior studies have proposed various approaches to tackle the instability arising from ill-defined principal components. The studies in [8] and [23] proposed a heuristic principled subspace-based approach to PCA based on the hypothesis that a subspace defined by a group of principal components may retain better stability relative to individual principal components if the eigenvalues corresponding to the said subspace are well-separated from the adjacent ones. PCA has also been studied with structural constraints on the covariance matrix (such as, sparsity [24] and diagonal structure [25]), where the said constraints enable improved robustness of learning outcomes to close eigenvalues.

Furthermore, implementation of PCA requires explicit computation of the eigendecomposition of the sample covariance matrix, an $\mathcal{O}(m^3)$ operation that becomes computationally prohibitive for high-dimensional datasets. In this context, existing studies have proposed various computationally efficient approaches to implementing PCA, including identification of principal components under sparsity constraints [26] or rank of the covariance matrix [27].

Finally, the conventional PCA implementations lack both induction and transferability. For instance, changes in the dimensionality of the dataset necessitate a complete recomputation of the principal components, limiting its utility in dynamic or large-scale applications (for instance, datasets with multiple spatial scales; see Fig. 6). In contrast to the prior studies that provide specialized solutions proposed to specific challenges faced by PCA, this article provides a holistic conceptual update to PCA through the lens of graph treatment of covariance matrix and GSP-driven information processing.

GSP-driven Implementation of PCA

Principles of GSP have bridged signal processing insights and mathematical theory over graphs with the empirical successes of GNNs [13]. A unique operational and conceptual perspective to PCA is obtained by treating PCA as a graph.

Covariance matrix as a graph. Formally, *a graph* $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{W})$ is a triplet with a set of m nodes $\mathcal{V} = \{1, \ldots, m\}$, a set of undirected edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and an weight function $\mathcal{W} : \mathcal{E} \mapsto \mathbb{R}$. Furthermore, the graph topology can be compactly represented using a symmetric matrix of size $m \times m$, which encodes the edge and weight information. Within this framework, the constituent features of the m-dimensional

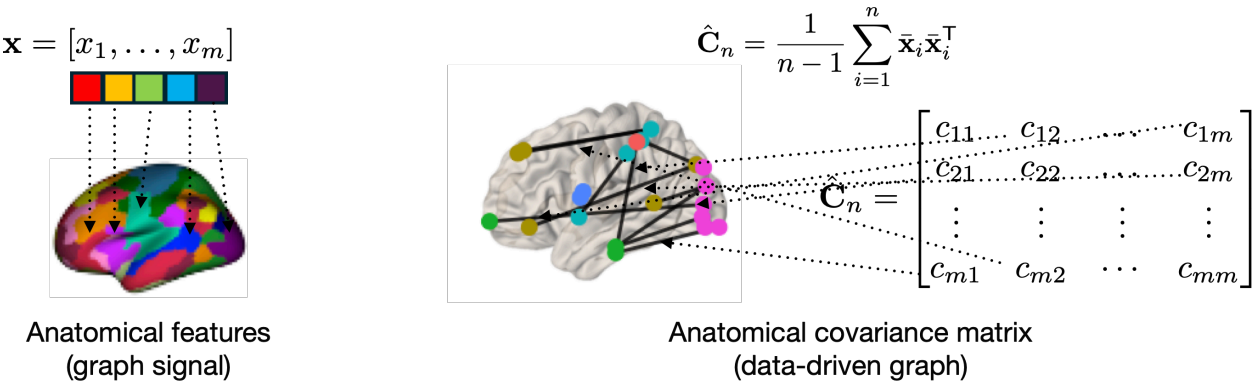

Fig. 2. A graph perspective to covariance matrix in the context of neuroimaging data. The data vector $\mathbf{x}$ represents anatomical information across the brain as it consists of readings or measurements at distinct brain regions. The non-diagonal entries of the corresponding (anatomical) covariance matrix represent the extent of linear coherence between the measurements at a given pair of brain regions. Thus, the covariance matrix admits a graph interpretation over the brain surface, with the nodes being different brain regions and the edges representing the coherence (for example, functional or anatomical) between them.

data vector $\mathbf{x}$ can be considered as 'nodes' of a graph, whose graphical structure is represented by the covariance matrix $\hat{\mathbf{C}}_n$ and $\mathbf{x}$ being the signal over the graph (graph signal). Figure 2 provides a practical example where the graphical perspective of covariance matrix is meaningful in a network neuroscience context.

Graph Fourier Transform and PCA. The graph Fourier transform (GFT) is the fundamental tool for spectral analysis of graph signals over a given graph structure. GFT of $\mathbf{x}$ over $\hat{\mathbf{C}}_n$ leverages the eigendecomposition in (2) and is given by

$$\breve{\mathbf{x}} = \hat{\mathbf{U}}^{\mathsf{T}}\mathbf{x}\,. \tag{6}$$

Clearly, the GFT in (6) reconciles with the PCA transform in (3). In the context of GSP, the eigenvalues of $\hat{\mathbf{C}}_n$ are mathematical equivalent of the notion of graph frequencies in GSP [16]. Thus, the i-th entry of $\breve{\mathbf{x}}$, i.e., $[\breve{\mathbf{x}}]_i$, represents the i-th Fourier coefficient corresponding to eigenvalue $\hat{\lambda}_i$. In an ML context, PCA-driven learning models learn the relative importance of individual eigenvectors of $\hat{\mathbf{C}}_n$. For instance, the scalar output of the PCA-based regression model with a scalar response in Fig. 3 is characterized by

$$\hat{y} = \sum_{i=1}^{m} \beta_i [\breve{x}]_i + \beta_0\,, \tag{7}$$

$$= \mathbb{1}^{\top}\mathcal{B}\breve{x} + \beta_0\,, \tag{8}$$

where $\mathcal{B} = \mathsf{diag}(\beta_1, \ldots, \beta_m)$ is the diagonal matrix of learned coefficients and $\beta_0 \in \mathbb{R}$ is the bias term learned during training. In the GSP context, the operation $\mathcal{B}\breve{x}$ in the implementation of PCA-driven

regression model in (8) is equivalent to implementing an 'ideal' spectral filter, where the filter coefficients are given by the diagonal elements in $\mathcal{B}$.

Achieving Spectral Filtering with Polynomial Operations. Spectral filtering in the spirit of $\mathcal{B}\breve{x}$ can be implemented via polynomial operations over $\hat{\mathbf{C}}_n$. Consider the operation

$$\mathbf{z} = h_0 + h_1 \hat{\mathbf{C}}_n \mathbf{x} \,, \tag{9}$$

where $h_0 \in \mathbb{R}$ and $h_1 \in \mathbb{R}$ are scalars. By leveraging the eigendecomposition of $\hat{\mathbf{C}}_n$, we can rewrite $\mathbf{z}$ in (9) as

$$\mathbf{z} = h_0 + h_1 \hat{\mathbf{U}} \mathbf{\Lambda} \hat{\mathbf{U}}^\top \mathbf{x} \tag{10}$$

$$= \hat{\mathbf{U}}(h_0 \hat{\mathbf{\Lambda}}^0) \hat{\mathbf{U}}^\top \mathbf{x} + \hat{\mathbf{U}}(h_1 \hat{\mathbf{\Lambda}}) \hat{\mathbf{U}}^\top \mathbf{x} \tag{11}$$

$$= \hat{\mathbf{U}}(h_0 \hat{\mathbf{\Lambda}}^0 + h_1 \hat{\mathbf{\Lambda}}) \hat{\mathbf{U}}^\top \mathbf{x} \,. \tag{12}$$

Thus, the spectral analysis of the elementary operation $h_0 + h_1 \hat{\mathbf{C}}_n \mathbf{x}$ reveals the dependence of the output $\mathbf{z}$ in (9) on eigenvector $\hat{\mathbf{u}}_i$ dictated via the function $h(\hat{\lambda}_i) = h_0 + h_1 \hat{\lambda}_i$. An increase in the sophistication of $h(\hat{\lambda}_i)$ (or equivalently, the design of the spectral filter) can readily be achieved by defining a higher-order polynomial in $\hat{\mathbf{C}}_n$ and without the explicit eigendecomposition of $\hat{\mathbf{C}}_n$. Specifically, for some arbitrary $K \in \mathbb{Z}^+$, the spectral analysis of the operation

$$\mathbf{z} = \sum_{k=0}^{K} h_k \hat{\mathbf{C}}_n^k \mathbf{x} \tag{13}$$

yields

$$\mathbf{z} = \sum_{k=0}^{K} h_k \hat{\mathbf{U}} \hat{\mathbf{\Lambda}}^k \hat{\mathbf{U}}^\mathsf{T} \mathbf{x} \quad = \hat{\mathbf{U}} \sum_{k=0}^{K} h_k \hat{\mathbf{\Lambda}}^k \hat{\mathbf{U}}^\mathsf{T} \mathbf{x} \,. \tag{14}$$

Thus, the dependence of the output $\mathbf{z}$ in (13) on eigenvector $\hat{\mathbf{u}}_i$ is dictated by the filtering operation $h(\hat{\lambda}_i) = \sum_{k=0}^{K} h_k \hat{\lambda}_i^k$. The coefficients $\{h_k\}_{k=0}^K$ form the *learnable* parameters that are estimated from data. Thus, the relative importances of the principal components of $\hat{\mathbf{C}}_n$ to the output $\mathbf{z}$ in (13) is determined in a data-driven fashion (i.e., through functions $\{h(\hat{\lambda}_i)\}_{i=1}^m$). More specifically, the graph filter modifies the contribution of the i-th eigenvector of $\hat{\mathbf{C}}_n$ via the function $h : \mathbb{R} \mapsto \mathbb{R}$ evaluated on the eigenvalue λ_i. Furthermore, in a spirit similar to that of PCA-based regression model in (8), a learning model that achieves learning objectives by data-driven exploitation of principal components can be realized via

$$\hat{y} = \mathbb{1}^\top \left(\sum_{k=0}^{K} h_k \hat{\mathbf{C}}_n^k \mathbf{x} \right) \,. \tag{15}$$

The polynomial operation $\sum_{k=0}^{K} h_k \hat{\mathbf{C}}_n^k$ in (13) is formally referred to as the 'graph filter' operation in the GSP context [17] and denoted by $\mathbf{H}(\hat{\mathbf{C}}_n)$ subsequently. To emphasize on the specialized focus on the

covariance matrix, we use the taxonomy of 'coVariance filter' to denote graph filters implemented on covariance matrix as the graph.

In practice, the learning of coefficients $\{h_k\}_{k=0}^{K}$ in a supervised learning regime can be achieved via prevalent stochastic gradient descent that solves the optimization problem defined over a training set. The output of a covariance filter, i.e., $\mathbf{z} = \mathbf{H}(\hat{\mathbf{C}}_n)\mathbf{x}$ provides a data-driven representation built upon the covariance structure in $\hat{\mathbf{C}}_n$ through the filter coefficients $\mathbf{h}$. In this scenario, the objective of the *training* procedure is to determine the filter coefficients $\mathbf{h}$ that provide the best possible fit of $\mathbf{z}$ with some known ground truth (gauged through a pre-decided loss function, such as mean squared error for regression tasks or cross entropy loss for classification tasks) for all inputs $\mathbf{x}$ in the training set. The covariance structure $\hat{\mathbf{C}}_n$ is estimated from the training dataset. The number of filter coefficients K is a design choice, which can be determined through a hyperparameter optimization procedure [28]. Figure 3 illustrates the realization of a regression model with a coVariance filter and a readout function and its conceptual equivalence with a standard PCA-driven regression model.

Principles of GSP and GNNs have been the focus of recent tutorial treatments; see e.g., [12], [13]. Within this context, $h(\hat{\lambda}_i)$ is formally referred to as the *frequency response* of the graph filter $\mathbf{H}(\cdot)$ and the coefficients $\{h_k\}_{k=0}^{K}$ are referred to as the *filter taps*. The fundamental equivalence between PCA-driven information processing and coVariance filter-driven processing observation has wide implications, as it can be built upon to overcome the key limitations of PCA. Firstly, the polynomial implementation underlying the coVariance filter is computationally more efficient relative to conventional PCA approaches that rely on computationally expensive eigendecomposition operation on the covariance matrix. Consequently, the coVariance filter implementation overcomes instability of PCA-driven approaches stemming from unreliable estimates of the eigenspectrum (discussed in 'Stability of VNNs' section). Moreover, the covariance filter $\mathbf{H}(\cdot)$ is a scale-free model as its parameters can be used to process a dataset of any arbitrary dimensionality without any changes to its architecture. Specifically, the filter $\mathbf{H}(\cdot)$ is contingent on the filter taps $\{h_k\}_{k=0}^{K}$ and not the dimensionality of the covariance matrix. Therefore, coVariance filter provides an analytic framework to theoretically analyze the transference across multiscale datasets [29] (discussed in 'Transferability of VNNs').

From coVariance Filters to VNNs The representation capacity of a covariance filter is limited to learning *linear* mappings over $\mathbf{x}$ through $\mathbf{H}(\hat{\mathbf{C}}_n)$. The representation capacity herein can be extended to *non-linear* mappings by nesting the covariance filter within a point-wise non-linear activation function; forming a coVariance perceptron. Specifically, for a given non-linear activation function $\sigma(\cdot)$, input $\mathbf{x}$, a coVariance filter $\mathbf{H}(\hat{\mathbf{C}}_n) = \sum_{k=0}^{K} h_k \hat{\mathbf{C}}_n^k$ and its corresponding filter coefficient set $\mathcal{H}$, the coVariance perceptron is

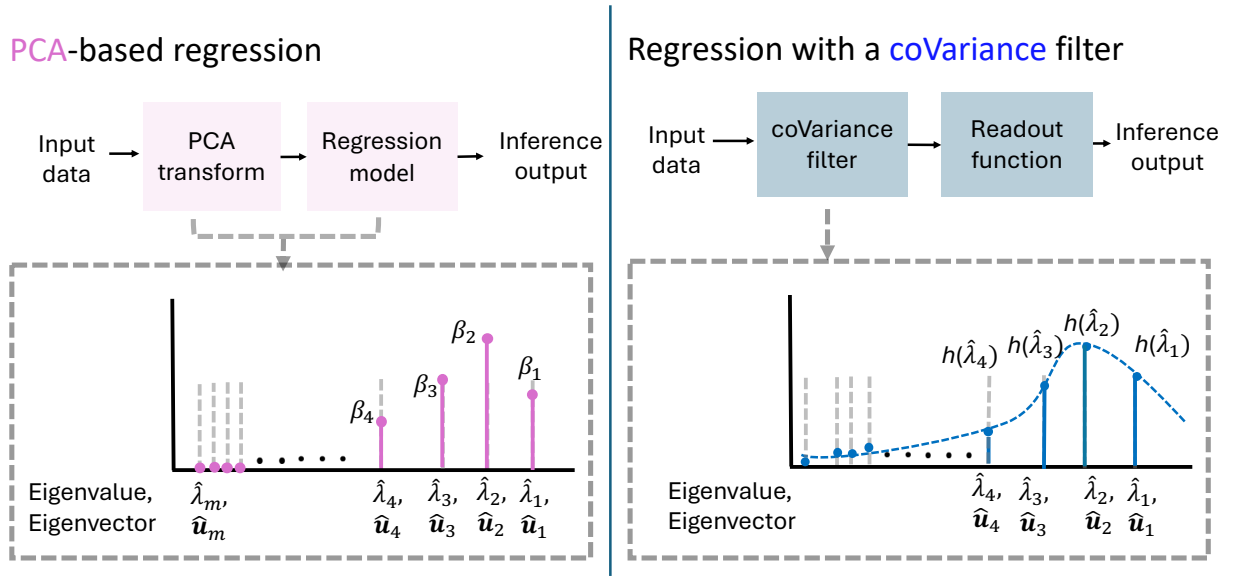

Fig. 3. PCA-driven regression model versus a coVariance filter-based regression model.

defined as

$$\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H}) \triangleq \sigma(\mathbf{H}(\hat{\mathbf{C}}_n)\mathbf{x}) \; . \tag{16}$$

Figure 4a illustrates a coVariance perceptron. A coVariance neural network (VNN) can be constructed by cascading multiple layers of coVariance perceptrons (see Fig. 4 for a two-layer VNN). This observation is formalized next.

Remark 1 (Multi-layer VNN): Consider an L-layer architecture formed by stacking L coVariance perceptrons. In this scenario, we denote the coVariance filter in layer ℓ by $\mathbf{H}_\ell(\hat{\mathbf{C}}_n)$ and its corresponding set of filter taps are given by $\mathcal{H}_\ell$. For a given pointwise nonlinear activation function $\sigma(\cdot)$, the relationship between the input $\mathbf{x}_{\ell-1}$ and the output $\mathbf{x}_\ell$ for the coVariance perceptron in the ℓ-th layer is given by

$$\mathbf{x}_\ell = \sigma(\mathbf{H}_\ell(\hat{\mathbf{C}}_n)\mathbf{x}_{\ell-1}) \quad \text{for} \quad \ell \in \{1, \ldots, L\}, \tag{17}$$

where $\mathbf{x}_0$ is the input $\mathbf{x}$. We refer to this L-layer architecture as an L-layer VNN.

The non-linear activation functions across different layers allow for non-linear transformations, thus, increasing the expressiveness of VNNs beyond linear mappings such as coVariance filters. In terms of architecture, VNNs are GNNs with covariance matrix as the graph shift operator [13]. Similar to GNNs and other deep learning models [14], [30], the representation power of VNNs can be increased by incorporating multiple parallel inputs and outputs per layer enabled by filter banks at every layer. In general, we use the notation $\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H})$ to denote the representation learnt over the input $\mathbf{x}$ by the VNN

with $\mathcal{H}$ as the set of all filter taps. For a supervised learning objective with a training dataset $\{\mathbf{x}_i, y_i\}_{i=1}^n$, the filter taps in $\mathcal{H}$ are chosen to minimize the training loss, i.e.,

$$\mathcal{H}_{\mathsf{opt}} = \min_{\mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(\Phi(\mathbf{x}_i; \hat{\mathbf{C}}_n, \mathcal{H}), y_i) \ , \tag{18}$$

where $\ell(\cdot)$ is some loss function that satisfies $\ell(\Phi(\mathbf{x}_i; \hat{\mathbf{C}}_n, \mathcal{H}), y_i) = 0$ iff $\Phi(\mathbf{x}_i; \hat{\mathbf{C}}_n, \mathcal{H}) = y_i$. Since the covariance matrix $\hat{\mathbf{C}}_n$ represents the covariance structure within the training set, we expect the mapping $\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H})$ to generalize well to data points $\mathbf{x}$ that are outside the training set but come from the same distribution as the training set.

Existing theoretical GNN studies typically assume an abstract graph representation and overlook the unique aspects stemming from a data-driven graph model provided by an empirical (sample) covariance matrix [8], [10]. Recent studies have undertaken rigorous theoretical analysis of VNNs [18]–[21], [29]; revealing unique theoretical and practical insights for deployment of GNNs in applications where covariance matrices inform the underlying graph structure. Furthermore, VNNs admit conceptual similarities with transformer architectures [31], where the self-attention mechanism in transformers can be interpreted as a generalized version of coVariance filters. This aspect has been discussed in 'Self-Attention in Transformers versus CoVariance Filter in VNNs'.

Self-Attention in Transformers versus CoVariance Filter in VNNs.

Self-attention is the central information processing mechanism in the prevalent transformer-based deep learning architectures [31]. Self-attention dynamically determines pairwise relevance based on feature representations, which motivates the broader interpretation of transformers as GNNs in prior studies [32]. Interestingly, self-attention also admits a principled interpretation in terms of covariance, revealing conceptual connections to coVariance filters. Self-attention leverages the following learnable linear projections of data matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$:

$$\mathbf{Q} = \mathbf{X}\mathbf{S}_{\mathbf{Q}}, \quad \mathbf{K} = \mathbf{X}\mathbf{S}_{\mathbf{K}}, \quad \mathbf{V} = \mathbf{X}\mathbf{S}_{\mathbf{V}} \ , \tag{19}$$

where $\mathbf{S}_{\mathbf{Q}} \in \mathbb{R}^{m \times q}$, $\mathbf{S}_{\mathbf{K}} \in \mathbb{R}^{m \times p}$, and $\mathbf{S}_{\mathbf{V}} \in \mathbb{R}^{m \times v}$ are learnable matrices. The resulting matrix $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are referred to as query, key, and value matrices, respectively. By leveraging $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, the attention matrix is computed as

$$\mathbf{A} = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{m}}\right) = \mathrm{softmax}\left(\frac{\mathbf{X}\mathbf{S}_{\mathbf{Q}}\mathbf{S}_{\mathbf{K}}^\top\mathbf{X}^\top}{\sqrt{m}}\right) = \mathrm{softmax}\left(\frac{\mathbf{X}\mathbf{M}\mathbf{X}^\top}{\sqrt{m}}\right) \ , \tag{20}$$

where $\mathbf{M} = \mathbf{S}_{\mathbf{Q}}\mathbf{S}_{\mathbf{K}}^\top$ and the $\mathrm{softmax}$ function implements row-wise normalization. Therefore, (20) reveals that evaluation of attention scores in $\mathbf{A}$ is contingent upon the term $\mathbf{X}\mathbf{M}\mathbf{X}^\top$, which can be

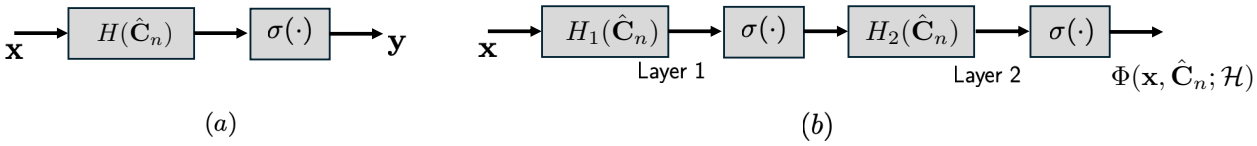

Fig. 4. (a) Perceptron formed by a coVariance filter $\mathbf{H}(\hat{\mathbf{C}}_n)$ and a pointwise non-linearity function $\sigma(\cdot)$. (b) A 2-layer VNN.

viewed as a generalization (achieved via the learnable matrix $\mathbf{M}$) of the standard covariance matrix employed in coVariance filters or VNNs. The output of the self-attention mechanism is given by

$$\mathbf{O} = \mathbf{AV} \ . \tag{21}$$

The output $\mathbf{O}$ takes the functional form of $\mathbf{O} = f(\mathsf{Cov}_{\mathbf{M}}(\mathbf{X}))\mathbf{X}$, where f is a non-linear normalization function and $\mathsf{Cov}_{\mathbf{M}} = \mathbf{XMX}^{\top}$. On the other hand, the output of the coVariance filter in (13) over the complete dataset admits the form $\mathbf{Z} = g(\mathsf{Cov}_{\mathbf{I}_m}(\mathbf{X}))\mathbf{X}$, where g is a linear function, $\mathbf{I}_m$ is the identity matrix of size $m \times m$ and therefore, $\mathsf{Cov}_{\mathbf{I}_m}(\mathbf{X})$ is the sample covariance matrix (as $\mathsf{Cov}_{\mathbf{M}}(\mathbf{X})$ reduces to $\hat{\mathbf{C}}_n$ when $\mathbf{M} = \mathbf{I}_m$). Thus, the self-attention mechanism processes a generalized form of covariance via weights in $\mathbf{M}$ to yield output. When $\mathbf{M}$ is set to Identity matrix, self-attention mechanism yields a coVariance filter with non-linear normalization. Similar arguments extend to yield the parallels between multi-head attention and a filter bank of coVariance filters.

Our discussion above reveals covariance as the common ground between the self-attention mechanism and coVariance filters. The superiority of attention-driven transformer architectures to other alternative learning models largely stems from their superior representation capacity, which comes at the cost of needing larger datasets for achieving similar performance as architectures with inbuilt specialized inductive biases (for instance, vision transformer versus a convolutional neural network [33]). Large-scale datasets may not be feasible in various real-world applications (for instance, in biomedical applications that require considerable resources to build huge datasets). For such scenarios with limited data, VNNs provide an effective deep learning solution that is intricately connected to the fundamental principles of transformers.

STABILITY OF VNNS

We reiterate from 'Challenges to Learning with PCA' that the reproducibility of inference outcomes through PCA-driven approaches is challenged by the statistical uncertainty in the estimation of principal components of $\hat{\mathbf{C}}_n$. Specifically, the eigenvalues $\{\hat{\lambda}_1, \ldots, \hat{\lambda}_m\}$ of $\hat{\mathbf{C}}_n$ are likely to be perturbed relative to the eigenvalues $\{\lambda_1, \ldots, \lambda_m\}$ of $\mathbf{C}$. Hence, for close eigenvalues of $\mathbf{C}$, the corresponding estimates

in $\hat{\mathbf{C}}_n$ may not maintain consistent ordering with a high likelihood (see (5)). Such considerations are relevant to learning with VNNs or PCA because we typically operate with $\hat{\mathbf{C}}_n$ rather than $\mathbf{C}$ in practical applications. Moreover, a traditional PCA-driven approach is highly vulnerable to irreproducibility when $\hat{\mathbf{C}}_n$ is perturbed relative to its estimate from the given training dataset (for instance, through a re-estimation of $\hat{\mathbf{C}}_n$ with more samples or in an independently collected dataset; see Fig. 1). In the context set above, the problem of establishing the stability of VNNs can be informally stated as follows.

Stability problem (Informal). *Given an input data sample* $\mathbf{x}$*, we aim to theoretically characterize the divergence between the representations* $\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H})$ *and* $\Phi(\mathbf{x}; \mathbf{C}, \mathcal{H})$*. If* $\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H})$ *and* $\Phi(\mathbf{x}; \mathbf{C}, \mathcal{H})$ *converge meaningfully (for instance, with increase in* n*), we can conclude that the VNN learns representations that are robust to the stochastic perturbations in* $\hat{\mathbf{C}}_n$ *relative to* $\mathbf{C}$*.*

CoVariance filter and VNN are discussed with respect to the sample covariance matrix $\hat{\mathbf{C}}_n$ estimated from the training dataset in 'From CoVariance Filters to VNNs'. In this context, the notion of stability of VNNs (or coVariance filters) is formalized through the impact of statistical uncertainty induced in the sample covariance matrix with respect to the ensemble covariance matrix on the coVariance filter output. To this end, the primary goal is to investigate the divergence $\|\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H}) - \Phi(\mathbf{x}; \mathbf{C}, \mathcal{H})\|$. If $\|\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H}) - \Phi(\mathbf{x}; \mathbf{C}, \mathcal{H})\|$ is controlled in some sense, we expect the learnable parameters $\mathcal{H}$ to meaningfully exploit the covariance structure in $\hat{\mathbf{C}}_n$, such that, a VNN with parameters $\mathcal{H}$ provides performance robust to the selection of $\hat{\mathbf{C}}_n$ and not overfit on the given specific instance of the training set. Thus, establishing stability within this context provides the critical theoretical guarantees on the robustness of representations learned by a VNN to stochastic perturbations in $\hat{\mathbf{C}}_n$ relative to $\mathbf{C}$, and consequently, the reproducibility of inference outcomes by coVariance filters or VNNs. Such theoretical guarantees are often missing from PCA-driven learning approaches. Theorem 1 characterizes the stability of coVariance filters and VNNs.

Theorem 1 (Stability of coVariance Filters and VNNs [18]): Consider a random vector $\mathbf{x} \in \mathbb{R}^{m \times 1}$, such that, $|\mathbf{x}| \leq 1$, and its corresponding ensemble covariance matrix $\mathbf{C} = \mathbb{E}[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{x} - \mathbb{E}[\mathbf{x}])^{\mathsf{T}}]$. For a sample covariance matrix $\hat{\mathbf{C}}_n$ formed using n i.i.d instances of $\mathbf{x}$, if the frequency response satisfies $|h(\lambda_i) - h(\lambda_j)| \leq \varsigma |\lambda_i - \lambda_j|$ for an appropriate $\varsigma > 0$, the following holds with a high likelihood:

$$\left\| \mathbf{H}(\hat{\mathbf{C}}_n) - \mathbf{H}(\mathbf{C}) \right\| \leq \frac{\varsigma}{n^{1/2-\varepsilon}} \mathcal{O}\left(\sqrt{m} + \frac{\|\mathbf{C}\| \sqrt{\log(nm)}}{\mu n} \right) = \alpha_n \,, \tag{22}$$

where α_n scales as $\mathcal{O}(1/n^{1/2-\varepsilon})$ for some $\varepsilon \in (0, 1/2)$ and a constant $\mu > 0$ that depends on the data distribution. Further, for a VNN $\Phi(\cdot; \cdot, \mathcal{H})$ with outputs per layer as F and L layers, if the pointwise

non-linearity $\sigma(\cdot)$ satisfies $|\sigma(a) - \sigma(b)| \leq |a - b|$, then,

$$\|\Phi(\mathbf{x}; \hat{\mathbf{C}}_n, \mathcal{H}) - \Phi(\mathbf{x}; \mathbf{C}, \mathcal{H})\| \leq LF^{L-1}\alpha_n \, . \tag{23}$$

The right-hand side term in (22) and the condition $|h(\lambda_i) - h(\lambda_j)| \leq \varsigma|\lambda_i - \lambda_j|$ are products of the analysis of the finite sample size effect-driven perturbations in $\hat{\mathbf{C}}_n$ and its eigenvectors with respect to that in $\mathbf{C}$. From Theorem 1, the divergence $\|\mathbf{H}(\hat{\mathbf{C}}_n) - \mathbf{H}(\mathbf{C})\|$ decays with the number of samples n at least at the rate of $1/n^{\frac{1}{2}-\varepsilon}$. Thus, the representations learned by VNN with $\hat{\mathbf{C}}_n$ as the covariance matrix converges with that learned by the VNN with $\mathbf{C}$ and same parameters $\mathcal{H}$.

The cost for the stability in (22) is also apparent from Theorem 1. Specifically, for a given eigenvalue λ_i, the response of the filter for any eigenvalue $\lambda_j, j \neq i$ becomes closer to $h(\lambda_i)$ with decrease in $|\lambda_i - \lambda_j|$. Recall from the perturbation theory of eigenvectors and eigenvalues of sample covariance matrices that the sample-based estimates of the eigenspaces corresponding to eigenvalues λ_i and λ_j become harder to distinguish as $|\lambda_i - \lambda_j|$ decreases [10]. Therefore, the coVariance filter that satisfies (22) sacrifices discriminability between close eigenvalues to preserve its stability with respect to the statistical uncertainty inherent in the sample covariance matrix. Specifically, the variation between frequency responses of the coVariance filter for any two eigenvalues λ_i and λ_j in Fig. 3, i.e., $|h(\lambda_i) - h(\lambda_j)|$ is governed by the difference $|\lambda_i - \lambda_j|$.

In contrast to coVariance filters, the VNNs incorporate non-linear activation functions within their information processing structure. Similar to GNNs [14], we expect the VNNs to retain the discriminability between close eigenvalues due to the demodulation impact of non-linear activation function. Case Study 1 provides an empirical study that corroborates the stability of VNN outcomes on a linear regression task while demonstrating the potential stability of PCA-driven models.

Sparse VNNs

The sample covariance matrix $\hat{\mathbf{C}}_n$ often contains spurious correlations and finite-sample estimation errors that produce inaccurate principal directions and result in a dense estimate. This hinders the estimation quality if the true covariance is sparse, and is detrimental for computational efficiency. While sparse PCA is a traditional remedy for improving estimation quality and reducing computation [34], [35], it suffers from the same instabilities as standard PCA. To address these limitations, Sparse VNNs (SVNNs) [19] integrate sparse covariance estimators directly into the VNN framework. If the true covariance is known or expected to be sparse, SVNNs employ hard thresholding, i.e., a function $\eta(\cdot)$ that

acts on each covariance entry $\hat{c}_{ij}$ as

$$\eta(\hat{\mathbf{C}}_n)_{ij} = \begin{cases} \hat{c}_{ij} & \text{if } |\hat{c}_{ij}| > \frac{\tau}{\sqrt{n}}, \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

That is, hard thresholding sets to zero the elements below the threshold. This covariance estimator admits the following stability analysis.

Theorem 2 (Stability of deterministic sparse covariance filters [19]): Consider a random vector $\mathbf{x} \in \mathbb{R}^{m\times 1}$, such that, $|\mathbf{x}| \leq 1$, and its corresponding ensemble covariance matrix $\mathbf{C} = \mathbb{E}[(\mathbf{x}-\mathbb{E}[\mathbf{x}])(\mathbf{x}-\mathbb{E}[\mathbf{x}])^{\mathsf{T}}]$ with at most c_0 non-zero elements per row. Consider also a sample covariance matrix $\hat{\mathbf{C}}_n$ formed using n i.i.d instances of $\mathbf{x}$, and let the frequency response satisfy $|h(\lambda_i) - h(\lambda_j)| \leq \varsigma|\lambda_i - \lambda_j|$ for an appropriate $\varsigma > 0$. Consider the application to the sample covariance of a hard-thresholding function $\eta(\cdot)$ with threshold $\tau = M'\sqrt{\log(m)}$ and M' a large enough constant. Let the eigenvalues of the true covariance $\{\lambda_i\}_{i=1}^N$ be all distinct and strictly positive. Then, the following holds with high probability:

$$\|\mathbf{H}(\eta(\hat{\mathbf{C}}_n)) - \mathbf{H}(\mathbf{C})\| \leq \mathcal{O}\left(\frac{\varsigma c_0 m\sqrt{\log m}}{n^{1/2}}\right). \tag{25}$$

Theorem 2 shows that the main takeaways of the stability analysis for covariance filters in Theorem 1 also apply to thresholded covariances. In particular, the bound decreases with the number of samples n, increases with the sample dimensions m and with the constant ς. Differently from standard covariance filters, however, the bound in (25) depends on the number of non-zero elements per row c_0 instead of the spectral norm of the covariance $\|\mathbf{C}\|$, which tightens it for sparse covariances. These considerations extend to sparse VNNs using (23).

However, the assumption of a sparse true covariance might not hold in practice, limiting the applicability and the validity of the stability results for soft thresholding. Therefore, SVNNs also consider stochastic sparsification, an approach where covariance values $\hat{c}_{ij}$ are dropped with probabilities p_{ij} that can be specified based on the application needs. For example, probabilities can be set to achieve a desired sparsification level, which guarantees a desired computational efficiency improvement, but might lose relevant covariance information. This results in the following stability result.

Theorem 3 (Stability of deterministic sparse covariance filters [19]): Consider a random vector $\mathbf{x} \in \mathbb{R}^{m\times 1}$, such that, $|\mathbf{x}| \leq 1$, and its corresponding ensemble covariance matrix $\mathbf{C} = \mathbb{E}[(\mathbf{x}-\mathbb{E}[\mathbf{x}])(\mathbf{x}-\mathbb{E}[\mathbf{x}])^{\mathsf{T}}]$ with at most c_0 non-zero elements per row. Consider also a sample covariance matrix $\hat{\mathbf{C}}_n$ formed using n i.i.d instances of $\mathbf{x}$, and let the frequency response be generalized integral Lipschitz with constant ς

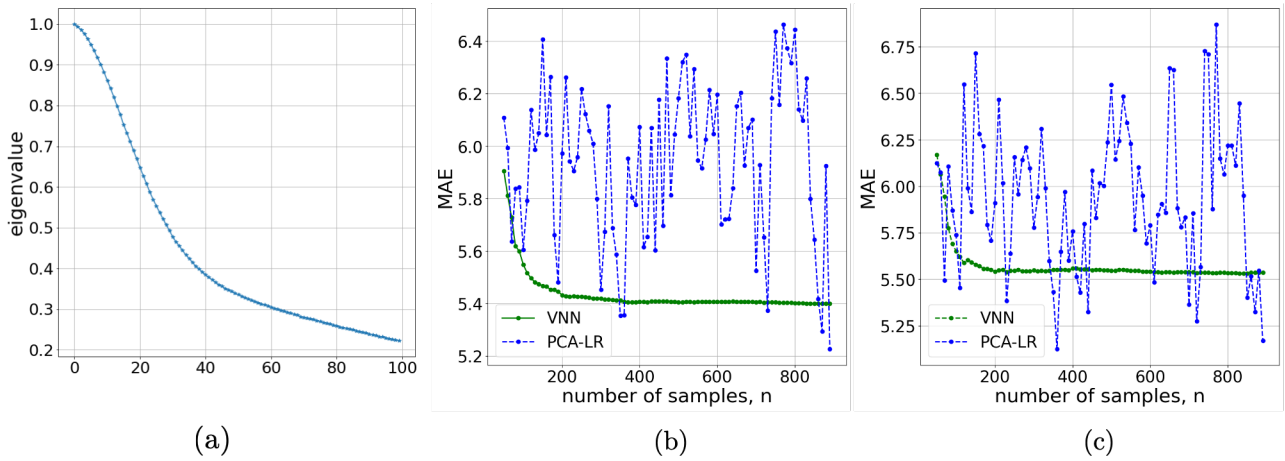

Fig. 5. Stability of VNNs on linear regression problem (tail = 0.7). (a) Eigenvalues of the covariance matrix $\hat{\mathbf{C}}_{900}$. (b) Variations in mean MAE performances over the training set with respect to perturbations in the sample covariance matrix that the models were trained on, i.e., $\hat{\mathbf{C}}_{900}$. (c) Variations in mean MAE performances over the test set with respect to perturbations in the sample covariance matrix that the models were trained on, i.e., $\hat{\mathbf{C}}_{900}$.

(see [19] for a formal definition). Consider applying to the sample covariance a stochastic thresholding function $\eta(\cdot)$ with probabilities p_{ij}. Then, the following holds with high probability

$$\mathbb{E}\left[\left\|\mathbf{H}(\eta(\hat{\mathbf{C}}_n)) - \mathbf{H}(\mathbf{C})\right\|^2\right] \leq \alpha_n^2 + m\varsigma^2 \sum_{i=1}^{m}\sum_{j=1}^{m} \hat{c}_{ij}^2(1-p_{ij}), \tag{26}$$

where α_n is defined in (22).

In this case, SVNNs' stability is affected by two sources of error: the finite-sample estimation α_n, which decreases with the number of samples, and the sparsification, which depends on the magnitude and quantity of covariance elements dropped. This indicates that removing large covariances $\hat{c}_{ij}^2$ might result in lower stability as relevant information is lost, which identifies a sparsity-stability tradeoff that can be controlled by the dropping probabilities p_{ij}. Moreover, this bound extends to complete sparse VNN architectures by applying (23).

Case Study 1. Stability of VNN outcomes on regression task.

In this case study, we investigated the stability of VNNs relative to PCA-driven models on the regression task. For this purpose, we generated random linear regression problems using the routine `sklearn.datasets.make_regression` in Python, which allows us to specify the number of informative features; effective rank of the input dataset, i.e., the approximate number of singular vectors required to explain most of the input data by linear combinations; tail parameter which is the relative importance of the fat noisy tail of the singular values profile; and noise. In our experiments,

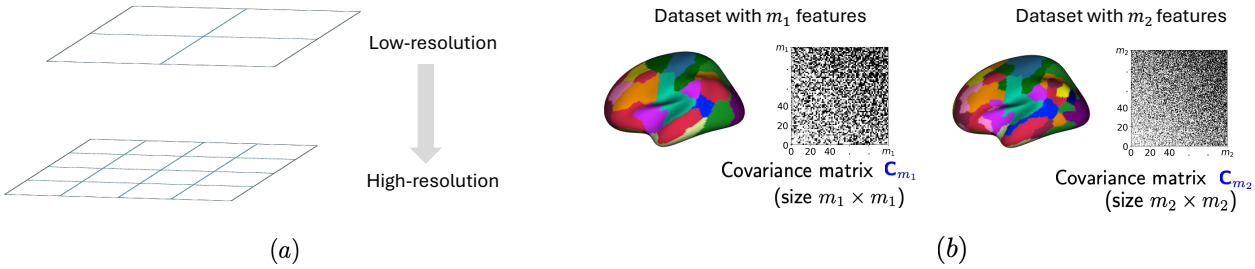

Fig. 6. (a) Abstract representation of a multiscale organization of a dataset. (b) Multiscale neuroimaging dataset illustrating partition of the brain surface at m_1 or m_2 resolutions with corresponding true covariance matrices $\mathbf{C}_{m_1}$ and $\mathbf{C}_{m_2}$.

we set the dimensionality of the input data to be $m = 100$, the dimension of the response to be 1, number of samples $n = 1000$, number of informative features to be 20, effective rank of the input dataset to be 25 and noise to be distributed according to $\mathcal{N}(0, 3)$. Furthermore, the tail parameter was set to 0.7. The spread of the eigenvalues of the covariance matrix is shown in Fig. 5 a.

The dataset was split into a $90/10$ train/test split, and generated sample covariance matrix $\hat{\mathbf{C}}_{900}$. Next, we deployed VNN and PCA-LR (a PCA-driven linear regression model) for the prediction task. The VNN consisted of 2 layers with 2 filter taps each, a filter bank of 13 m-dimensional outputs per layer, and a readout layer that calculated the unweighted mean of the outputs at the last layer to form an estimate for the response. The PCA-regression pipeline consisted of two steps: i) we first identified the principal components using the eigendecomposition of $\hat{\mathbf{C}}_{900}$; and then, ii) to maintain consistency with VNN, transformed the training set used for VNN training to fit to the corresponding response data using a linear regression model. The optimal number of principal components in the PCA-regression pipeline was selected through a 10-fold cross-validation procedure on the training set, repeated 5 times.

Using different permutations of the training set, we obtained 100 nominal models for VNN and $\mathsf{PCA-IR}$. Figure 5 b) plots the variations in mean MAE performances of the nominal models with respect to perturbations in the sample covariance matrix, with the last data point corresponding to $n = 900$, i.e., $\hat{\mathbf{C}}_{900}$. When $\hat{\mathbf{C}}_{900}$ is replaced with $\hat{\mathbf{C}}_{n'}$ for any $n' \in [5, 899]$, our experiments showed that VNN performance was stable, but significant randomness was induced into the performance of PCA-LR model. A similar phenomenon was observed for the performance on the test set in Fig. 5c).

TRANSFERABILITY OF VNNS

In the multiscale setting with spatial resolutions m_1 and m_2, the dataset with m_1 features and n data samples is denoted by $\mathbf{X}_n^{m_1} \in \mathbb{R}^{n \times m_1}$ and that with m_2 features and n data samples is denoted

by $\mathbf{X}_n^{m_2} \in \mathbb{R}^{n \times m_2}$. Both datasets are assumed to represent the same information at distinct resolutions. See Fig. 6a for an abstract representation of the multiscale setting considered here. Figure 6b illustrates the multiscale organization in the setting of a neuroimaging dataset curated according to two distinct resolutions of a brain atlas. The corresponding true covariance matrices for the datasets with m_1 and m_2 features are denoted by $\mathbf{C}_{m_1}$ and $\mathbf{C}_{m_2}$, respectively. For conciseness in exposition and avoiding repetition of arguments, we have focused our discussions in this section on the true covariance matrices as opposed to sample covariance matrices in the previous sections. The problem statement of the transference of VNNs across multiscale datasets is informally stated next.

Transference problem (Informal). *Given a data point $\mathbf{x}_{m_1}$ from the dataset with m_1 features and associated covariance matrix $\mathbf{C}_{m_1}$, and another data point $\mathbf{x}_{m_2}$ from the dataset with m_2 features and associated covariance matrix $\mathbf{C}_{m_2}$, we aim to characterize the operator distance between the representations $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$. If $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$ converge in some sense, we can conclude that the VNN with parameters $\mathcal{H}$ is transferable between two datasets consisting of m_1 and m_2 features.*

The above problem involves comparing the representations $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$, which are of different dimensionalities and hence, this comparison is not natural. This comparison is facilitated theoretically by the mapping of the datasets and associated representations to the same continuous domain [29]. As an example of a scenario where such a mapping could be valid, consider two neuroimaging datasets curated according to different brain atlases (and hence, having different number of features) but capturing brain activity across the whole brain. In this context, the scope of information for both datasets is limited to the same brain surface. Since the datasets with m_1 and m_2 features can be imagined as discrete samples of the same continuum (brain surface), we can compare them mathematically by mapping them to the same continuous domain.

In previous works [29], the following mapping of data to the continuous domain has been considered: Given an m-dimensional vector $\mathbf{x} = [x_1, \ldots, x_m]$, we can define a continuous representation of $\mathbf{x}$ as a function $y_{\mathbf{x}} : [0, 1] \mapsto \mathbb{R}$, such that, $y_{\mathbf{x}}(u) = x_i$ for $u \in \mathcal{U}_i$, where $\mathcal{U}_i$ is a predefined interval associated with the i-th element of $\mathbf{x}$, and proportional to the relative *variance* of a feature. Similarly, we can map a covariance matrix $\mathbf{C}_m$ to a compact set $[0, 1]^2$ using the mapping $\mathbf{W}_{\mathbf{C}_m} : [0, 1]^2 \mapsto \mathbb{R}$, where we have $\mathbf{W}_{\mathbf{C}_m}(u, v) = [\mathbf{C}_m]_{ij}$ for $u \in \mathcal{U}_i$ and $v \in \mathcal{U}_j$. A pictorial illustration of $y_{\mathbf{x}}$ and $\mathbf{W}_{\mathbf{C}}$ for covariance matrix $\mathbf{C}$ is included in Fig. 7.

The consequence of having a continuous approximation is that, given the setting with multiscale data points $\mathbf{x}_{m_1}$ and $\mathbf{x}_{m_2}$, the closeness of continuous representations $y_{\mathbf{x}_{m_1}}$ and $y_{\mathbf{x}_{m_2}}$ can be used as a metric to assess the similarity between them. This observation also extends to the theoretical comparisons between

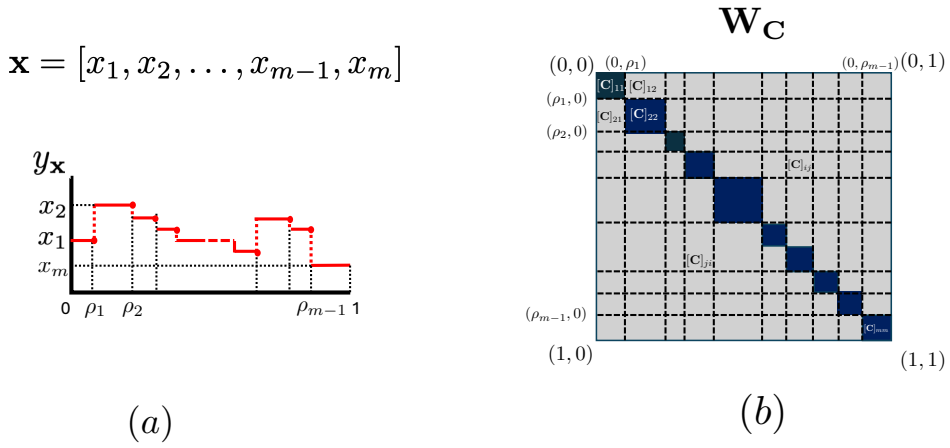

Fig. 7. Continuous approximations of (a) m-dimensional discrete data $\mathbf{x}$ in the interval $[0,1]$ and (b) an $m \times m$ covariance matrix in the space $[0,1]^2$.

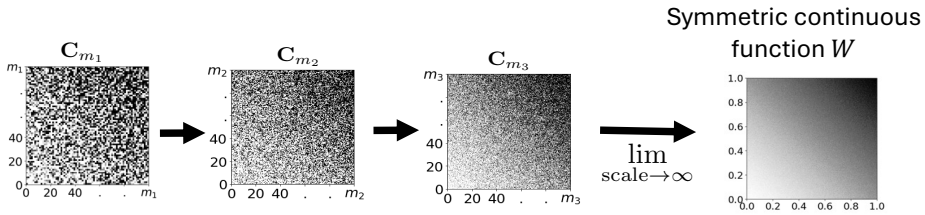

Fig. 8. Convergence of a sequence of covariance matrices to a symmetric, continuous function in the limit of scale (number of features) approaching ∞.

matrices $\mathbf{C}_{m_1}$ and $\mathbf{C}_{m_2}$ or representations $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$. Hence, the transference problem for VNNs can be formalized mathematically as follows.

Transference problem with continuous approximations: Consider two representations $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$ derived by the same covariance filter coefficients $\mathcal{H}$ on datasets with m_1 and m_2 features, respectively. If we have the following conditions: (a) the continuous approximations of inputs $\mathbf{x}_{m_1}$ and $\mathbf{x}_{m_2}$ are close, i.e., $\|y_{\mathbf{x}_{m_1}} - y_{\mathbf{x}_{m_2}}\|$ is bounded; and (b) the continuous approximations of covariance matrices $\mathbf{C}_{m_1}$ and $\mathbf{C}_{m_2}$ are close, i.e., $\|\mathbf{W}_{\mathbf{C}_{m_1}} - \mathbf{W}_{\mathbf{C}_{m_2}}\|$ is bounded; we aim to characterize the closeness between the continuous approximations of representations $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$, i.e., find $\vartheta > 0$, such that, $\|y_{\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})} - y_{\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})}\| \leq \vartheta$.

Establishing the transference problem as above provides a novel perspective to the domain of transfer learning in multiscale settings, where VNNs provide novel insights regarding the interplay between the redundancy in information across multiple scales and the quality of transference of VNNs. Such theoretical insights are simply infeasible with prevalent deep learning networks in this context [36], [37].

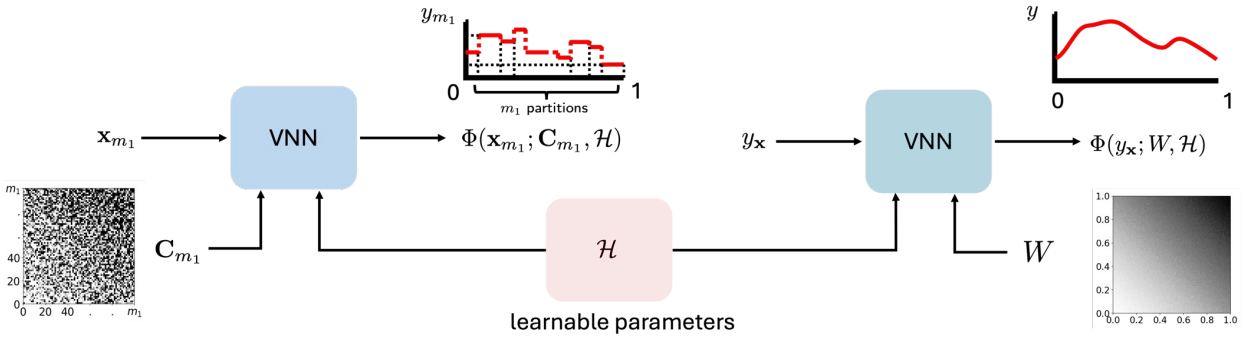

Fig. 9. Comparison of representations learned by VNN from a data sample with finite scale and corresponding data sample in the limit of infinite scale.

Theorem 4 (Transference of VNNs (Informal) [29]): Consider two datasets of m_1 and m_2 features and a VNN $\Phi(\cdot;\cdot,\mathcal{H})$ consisting of L layers and F outputs per layer. If the continuous approximations $\mathbf{W}_{\mathbf{C}_{m_1}}$ and $\mathbf{W}_{\mathbf{C}_{m_2}}$ are close and part of a converging sequence, and the continuous approximations $y_{\mathbf{x}_{m_1}}$ and $y_{\mathbf{x}_{m_2}}$ are close, then the continuous approximations of the representations $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$ converge as

$$\|y_{m_1} - y_{m_2}\|_2 = \mathcal{O}\Big(\frac{1}{m_1^{3\zeta/2-1}} + \frac{1}{m_2^{3\zeta/2-1}}\Big) \quad \text{for some constant } \zeta \in (2/3, 1]. \tag{27}$$

Theorem 4 implies that continuous representations for the VNN outputs $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$ converge with increase in m_1 and m_2. As an extension, we expect the measures of central tendency (e.g., mean, median) of the VNN outputs in the multiscale setting to converge as well. Hence, if the VNN provides the inference output with the unweighted mean as the readout function, we expect the statistical outcomes derived from $\Phi(\mathbf{x}_{m_1}; \mathbf{C}_{m_1}, \mathcal{H})$ and $\Phi(\mathbf{x}_{m_2}; \mathbf{C}_{m_2}, \mathcal{H})$ to be close or consistent and this convergence to be stronger for large m_1 and m_2. Key technical components that lead to the results in Theorem 4 have been discussed in 'Analytical components behind the theoretical analysis of transferability of VNNs'. The experiments in Case Study 2 demonstrated the successful transference of VNN models across multiscale neuroimaging datasets curated according to different scales of Schaefer's brain atlas [38].

Analytical components behind the theoretical analysis of transferability of VNNs.
The theoretical approach to characterizing transference focuses on defining the *limit of the dataset and covariance network in the asymptote of infinite number of features or dimensionalities*. Assuming that such limits exist, we will be able to characterize the notion of convergence between the datasets at finite scales. The existence of such a limit is intuitively expected for various applications. An

example of brain activity being captured on the continuum of the brain surface at different scales in neuroimaging datasets was discussed above. Furthermore, the geospatial datasets capture information about a physical spatial phenomenon, such as ocean temperature or soil properties, via a strategic location of sensors [39]. Figure 8 provides a figurative illustration of such a sequence, where the covariance matrices formed by datasets with increasing scale or number of features converge to a continuous function $\mathbf{W}$ as the scale approaches infinity.

The continuous representations of graph signals and graphs have previously been leveraged to study transferability of GNNs under the domain of graphon information processing [40]. Specifically, GNNs can be transferable between graphs belonging to a converging sequence if the graphs in this sequence converge to a limit object called *graphon* as the number of nodes approaches infinity [41]. Similarly, the convergence of covariance matrices towards a graphon limit [41] and analysis of VNNs within the regime of graphon signal processing [40] is instrumental to establishing Theorem 4. Graphons are the limits of *dense* graphs (i.e., graphs with number of edges of the order $\Theta(m^2)$) [42] and hence, appropriate to study limits of covariance matrices that are typically dense. The definition of a graphon is provided in Definition 1.

Definition 1 (Graphon): A graphon is a bounded, symmetric, measurable function $\mathbf{W} : [0,1]^2 \mapsto [-1,1]$.

Subsequently, the theoretical characterization of transference of VNNs hinges on the following analytical steps:

Analytical component (a) *VNNs in the continuous domain*: Defining the VNN in the continuous domain is contingent on demonstrating that a coVariance filter $\mathbf{H}(\mathbf{C}_m)$ can be equivalently represented using convolution operations over continuous approximations $\mathbf{W}_{\mathbf{C}_m}$ of covariance matrix $\mathbf{C}_m$ and $y_{\mathbf{x}_m}$ of input sample $\mathbf{x}_m$. The operation $\mathbf{C}_m\mathbf{x}_m$ is fundamental to the convolution operation in $\mathbf{H}(\mathbf{C}_m)\mathbf{x}$ and therefore, we first demonstrate how its continuous equivalent will be evaluated for the tabular setting. For $\mathbf{s} = \mathbf{C}_m\mathbf{x}_m$, the i-th element of $\mathbf{s}$ is $[\mathbf{s}]_i = \sum_{j=0}^{m}[\mathbf{C}_m]_{ij}[\mathbf{x}]_j$. Thus, $[\mathbf{s}]_i$ is a linear combination of elements in $\mathbf{x}$ according to the i-th row of $\mathbf{C}_m$. In the continuous space, we can equivalently represent $\mathbf{s}$ as $y_{\mathbf{s}}(u) = \int_0^1 \mathbf{W}_{\mathbf{C}_m}(u,v)y_{\mathbf{x}}(v)\mathrm{d}v$. This observation can be readily extended to define the continuous equivalent of a covariance filter. This is feasible because we can write the entity $\mathbf{C}_m^k\mathbf{x}$ in $\mathbf{H}(\mathbf{C})$ in a recursive form. Specifically, if we have $\mathbf{s}_k = \mathbf{C}_m^k\mathbf{x}$, then we can rewrite $\mathbf{s}_k$ as $\mathbf{s}_k = \mathbf{C}_m\mathbf{s}_{k-1}$, where $\mathbf{s}_0 = \mathbf{x}$. Thus, using the same reasoning that established the equivalence between $\mathbf{s}$ and $y_{\mathbf{s}}$, we conclude that the continuous representation $y_{\mathbf{s}_k}$ of $\mathbf{s}_k$ can be recovered via the following operation $y_{\mathbf{s}_k}(u) = \int_0^1 \mathbf{W}_{\mathbf{C}_m}(u,v)y_{\mathbf{s}_{k-1}}(v)\mathrm{d}v$. Since the coVariance filter output $\mathbf{z}$ is a weighted aggregation of the terms $\mathbf{s}_k$, we will be able to write its

continuous representation $y_{\mathbf{z}}$ as $y_{\mathbf{z}}(u) = \sum_{k=0}^{K} h_k y_{\mathbf{s}_k}(u)$. Using the mathematical steps leading up to this point, we have demonstrated that the continuous representation of the covariance filter output $\mathbf{z}$ can be recovered via the convolution operations over the continuous representation $\mathbf{W}_{\mathbf{C}_m}$. The extension of this observation to defining the representation $\Phi(\mathbf{x}_m; \mathbf{C}_m, \mathcal{H})$ for a VNN to its continuous representation $y_{\Phi(\mathbf{x}_m;\mathbf{C}_m,\mathcal{H})}$ is straightforward as it involves building a neural network architecture defined on covariance filters in the continuous domain [29].

Analytical component (b) *Convergence of the sequence of continuous approximations* $\{\mathbf{W}_{\mathbf{C}_m}\}_{m=m_0}^{\infty}$ *to limit* $\mathbf{W}$. The existence of the continuous limit $\mathbf{W}$ for a sequence of continuous approximations $\{\mathbf{W}_{\mathbf{C}_m}\}_{m=m_0}^{\infty}$ for covariance matrices from multiscale datasets (for some arbitrary initial scale m_0) is contingent on the appropriate construction of the continuous approximations $\mathbf{W}_{\mathbf{C}_m}$ and an appropriate measure of *distance* between them. Specifically, when (i) continuous approximations $\mathbf{W}_{\mathbf{C}_m}$ are constructed on the space $[0,1]^2$, such that, the *area* associated with a feature is proportional to its variance, and (ii) the *cut distance* between the successive continuous approximations relative to some metric follows a Cauchy sequence [41], we can leverage the graphon theory for weighted graphs to conclude that the sequence of the continuous approximations for covariance matrices for the multiscale datasets are indeed converging [29]. Intuitively, such a convergence of the continuous approximations across scales implies that the multiscale datasets contain only the redundant information across all scales.

Analytical component (c) *Spectral analysis of VNNs in continuous domain.* The comparison of representations learned by VNNs applied on datasets of different scales will be facilitated by spectral decomposition of the continuous approximations of their respective outputs. Spectral analysis of VNNs in the continuous domain will hinge on the continuous approximations $\mathbf{W}_{\mathbf{C}_m}$ and the graphon limit function $\mathbf{W}$ being *Hilbert-Schmidt operators* [43]. The implication of this assumption is that the continuous approximations $\mathbf{W}_{\mathbf{C}_m}$ and the limit function $\mathbf{W}$ will be continuous, compact and admit the properties (i) $\int_0^1 \int_0^1 \mathbf{W}(a,b)\mathrm{d}a\mathrm{d}b < \infty$ and $\int_0^1 \int_0^1 \mathbf{W}_{\mathbf{C}_m}(a,b)\mathrm{d}a\mathrm{d}b < \infty$, and (ii) eigendecompositions of the form: $\mathbf{W}(u,v) = \sum_{i\in\mathbb{Z}\backslash\{0\}} \eta_i \Gamma_i(u)\Gamma_i(v)$, where $\eta_i, \forall i \in \mathbb{Z}\backslash\{0\}$ are eigenvalues and Γ_i are the eigensignals of $\mathbf{W}$. Similar properties hold for continuous approximations $\mathbf{W}_{\mathbf{C}_m}$ under the assumption of them being Hilbert-Schmidt operators. The consequence of this discussion here is that we can analyze the output of a covariance filter defined on the continuous function $\mathbf{W}$ as $y(u) = \sum_{i\in\mathbb{Z}\backslash 0} \sum_{k=0}^{K} h_k \eta_i^k \Gamma_i(u) \int_0^1 \Gamma_i(v) x(v) \mathrm{d}v$ for an input function $x : [0,1] \mapsto \mathbb{R}$ and compare it mathematically with the spectral decomposition of the covariance filter defined on continuous approximation $\mathbf{W}_{\mathbf{C}_m}$ [29].

Case Study 2. Transference of VNNs across multiscale neuroimaging datasets.

In this case study, we demonstrate the transferability of VNN models across multiscale cortical thickness datasets curated according to different scales of 17-network Schaefer's brain atlas [38]. VNN model was trained on the 100-parcellations dataset to predict the chronological age of a healthy population. Transference of VNN was investigated by deploying the trained VNN model on the dataset from the same population that was curated according to the 200-parcellation version of Schaefer's brain atlas.

Datasets. We used the cortical thickness features derived from T1-weighted (T1w) MRI images collected from 3.0 Tesla MRI scanners for several publicly available datasets constituted by 2147 healthy individuals who were 18 years or older. These datasets include: (a) Cambridge Centre for Ageing and Neuroscience (CamCAN) dataset [44]; (b) Dallas Lifespan Brain Study (DLBS) https://fcon_1000.projects.nitrc.org/indi/retro/dlbs.html; (c) IXI dataset (https://brain-development.org/ixi-dataset/; and enhanced Nathan Kline Institute-Rockland Sample (eNKI) [45]. The demographics of these datasets are summarized in Table I. The T1w MRI images in these datasets were preprocessed via the open-source CAT12 pipeline [46] to yield cortical thickness measures for each individual. In particular, for every individual, we obtained cortical thickness measures curated according to 100 parcellations, 200 parcellations, and 400 parcellations versions of Schaefer's brain atlas [38]. The VNN was trained on the cortical thickness dataset from the population in Table I that was curated according to the 100 parcellation version of Schaefer's brain atlas.

Architecture and training. VNN was trained to predict the chronological age using 100 cortical thickness features across the cortex for the population of healthy individuals, described in Table I. The architecture consisted of 3-layers, with 2 filter taps each in the first and second layers, and 9 filter taps in the third layer. The width of the network was set to 35. The architecture parameters were selected based on the results of a hyperparameter search performed over 100 trials using the Optuna package [28]. The training procedure was based on optimizing the mean-squared-error (MSE) loss function. The learning rate for the Adam optimizer was 0.01. The model was trained on 90% of the dataset and tested on the remaining 10% to validate that it could infer information about chronological age on an unseen healthy population. The $90/10$ split of the dataset was determined randomly, and the constitution of the 10% unseen data is provided in the supplementary material. The model operated on the anatomical covariance matrix of size 100×100 estimated from the 100 cortical thickness features of the 90% split and normalized, such that its maximum eigenvalue was 1. Furthermore, the cortical thickness features were z-score normalized across the 90% split

and this normalization was applied to the 10% validation cohort. Based on 10 models trained on different permutations of the training set, NeuroVNN achieved an error performance of 9.24 ± 0.59 years with correlation between the ground truth and NeuroVNN estimates of chronological age being 0.79 ± 0.004 on the test set. On the complete dataset, NeuroVNN achieved similar performance; the error was 9.16 ± 0.26 years, and the correlation between the ground truth and NeuroVNN estimates of the chronological age was 0.796 ± 0.0032.

Results. Figure 10 demonstrates high similarity between the VNN-driven age predictions for the 100-dimensional dataset (that it was trained upon) and its outputs after being transferred to the 200 and 400 dimensional versions of the same dataset (Pearson's correlation > 0.97 for all scenarios).

Remarks. The results here have demonstrated the transferability of VNNs to datasets of different dimensionalities for the downstream tasks. This ability of VNN is remarkable as it demonstrates the ability of VNN to exploit the redundant information across datasets of different dimensionalities; adding a key perspective to the generalization of VNN. Prior works have exploited the transference of VNNs to demonstrate generalizations of anatomical patterns reflecting neurodegeneration across different neuroimaging datasets [47], [48].

TABLE I

DEMOGRAPHICS FOR DATASETS USED FOR TRAINING VNN IN CASE STUDY 2.

Dataset	n	Sex (m/f)	Age	
			range	mean ± s.d.
CamCAN	652	322/330	18.5-88.92	54.77 ± 18.6
DLBS	315	117/198	20.57 -89.11	54.62 ±20.09
IXI	182	88/94	20.16-81.94	47.44±16.7
eNKI	998	347/650*	18-85	47.35±17.71
Total	2147	874/1272*	18-89.11	50.68±18.62

* Sex information missing for 1 individual.

EXTENSIONS OF VNNS

The properties of VNNs have established them as a versatile foundation for covariance-based learning. Several recent developments have extended the framework to accommodate complex data structures [20], specialized estimators [21], [49], and diverse operational constraints such as label scarcity [50], often drawing powerful parallels with classical PCA variants and while investigating the impact of the finite-sample stability.

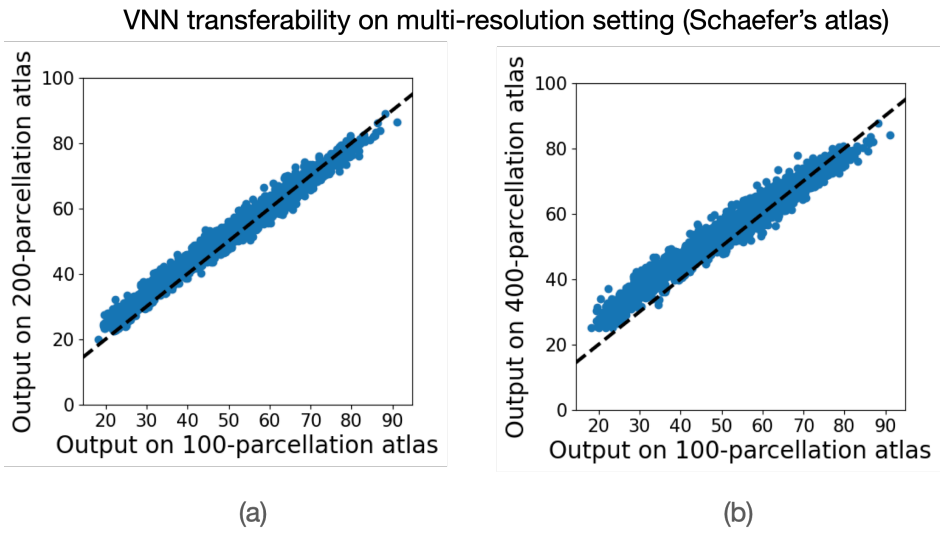

Fig. 10. Transferability of VNN for chronological age prediction task. VNN model was trained on a dataset curated according to the 100-parcellation version of Schaefer's brain atlas and transferred to other versions of the dataset curated according to the 200- and 400-parcellation versions of Schaefer's brain atlas.

Task-Based Covariance Estimation

Building on the foundations of VNNs, several extensions have been developed to address specific data structures or dependencies, often drawing parallels to specialized PCA variants. We describe here the extensions to temporal, biased, and noisy data.

While standard VNNs effectively capture spatial relationships, the SpatioTemporal VNN (STVNN) framework introduced in [20] is designed to account for the intrinsic temporal dependencies found in domains like neuroimaging, sensor networks, and finance. To achieve this, STVNNs modify the standard graph convolution operation by integrating a summation over a temporal window. Consider a dataset of temporal observations $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$. A spatiotemporal covariance filter with window size t applied to the n-th observation performs the operation

$$\mathbf{z} = \sum_{\tau=0}^{t-1} \sum_{k=0}^{K} h_{k\tau} \hat{\mathbf{C}}_n^k \mathbf{x}_{n-\tau}. \tag{28}$$

Since the learnable coefficients $h_{k\tau}$ depend both on the time lag and the shift order, this filter effectively captures spatiotemporal patterns. To support real-world streaming applications, STVNNs utilize online updates for both the covariance estimates and the filter coefficients. Under stationary conditions, this iterative approach guarantees convergence to the true underlying covariance; in non-stationary environments, it enables the model to adapt to distribution shifts. This formulation establishes a direct parallel with spatiotemporal and online PCA variants through the spectral processing of spatiotemporal

covariance information. By leveraging the inherent robustness of the VNN architecture, STVNNs provide a mathematically grounded framework that guarantees stability against finite-sample estimation errors.

In [21], VNNs are extended to the domain of algorithmic fairness, ensuring equitable performance across diverse demographic or categorical groups. Specifically, Fair VNNs (FVNNs) aim to achieve consistent accuracy on downstream tasks regardless of group membership, particularly when the task itself should be agnostic to such attributes. Standard sample covariance estimators often inherit and amplify data biases, resulting in PCA projections that favor some groups over others. Moreover, the stability issues intrinsic in PCA yield principal components that are more accurate for majority groups and more noisy for underrepresented ones, amplifying the existing biases. The FVNN paradigm addresses this problem with two improvements. First, FVNNs employ fair covariance estimators, that either reweight the contributions of different groups to compensate for sample size imbalances or apply transformations to decouple group dependencies from the covariance structure entirely. Second, FVNNs are trained to minimize an objective function that balances the downstream task performance and a bias mitigation penalty. This formulation effectively extends fair PCA methods by introducing learnable, fair weights, providing greater flexibility in capturing task-relevant information. Furthermore, FVNNs inherit the characteristic stability of the VNN architecture, remaining robust to finite-sample estimation errors within the fair covariance estimators themselves and mitigating unfair treatments related to covariance estimation errors.

Finally, the work in [49] considers the setting where observations are perturbed by outliers (e.g., interferences in sensor measurements) or missing values (e.g., temporary sensor failures). Such perturbations differ from the finite-sample estimation errors as they are significantly larger in magnitude, thus violating the assumptions behind VNNs' stability result and deteriorating their downstream task performance. To mitigate these effects, Robust VNNs (RVNNs) [49] estimate two covariance correction matrices that compensate for data corruption. Following the robust PCA data model [51], these corrections are modeled using one low-rank and one sparse prior, and they are learned end-to-end by minimizing a loss that balances downstream task accuracy with the structural priors of the correction matrices. In this way, RVNNs produce task-aware robust covariances that filter out large-scale perturbations while simultaneously leveraging label information. This integrated approach ensures that RVNNs remain stable in the presence of both standard finite-sample estimation errors and significant data corruption.

Covariance Scattering Transforms

VNNs are inherently supervised architectures that rely on training to estimate an appropriate set of coefficients, which generally requires labeled data for a downstream task. In many practical applications, however, data is abundant but labels are scarce, rendering standard VNNs difficult to deploy. To overcome

this limitation, Covariance Scattering Transforms (CSTs) [50] provide a framework for generating robust, covariance-aware representations without training. CSTs are based on covariance wavelets, filtering functions localized in both the spectral and spatial domains of the covariance matrix. These wavelets extract specific covariance patterns and are instantiated at different scales to capture shifted or modified versions of these patterns. CSTs consist of a cascade of covariance wavelets at different scales applied to the input and interleaved with non-linear activation functions. This multi-layer architecture produces data representations that are covariance-aware, expressive, untrained, and stable to finite-sample estimation errors, thus merging the untrained nature of PCA and the stability and expressivity of VNNs. Since CSTs' representations can significantly grow in dimension, a pruning mechanism is adopted to remove low-energy branches, speed up computation, and reduce memory consumption while maintaining stability and expressivity. Ultimately, the high-quality features produced by CSTs allow simple readout models to achieve strong downstream task performance even in low-data regimes or settings with minimal label availability.

Conclusions and Future Outlook

PCA has been widely adopted for data analysis and statistical inference in recent decades. The success of PCA stems from various factors, including minimal assumptions, model-agnostic characteristics, and interpretability tied to intuitive understanding of the covariance matrix. We began this tutorial by highlighting the critical conceptual and operational challenges faced by the conventional covariance matrix-based PCA approaches; broadly encompassing the lack of scalability (due to their dependence on explicit eigendecomposition of the covariance matrix) and lack of guarantees on the reproducibility of findings in finite data and multiscale data regimes. The aforementioned challenges appear prominently in modern data regimes characterized by high-dimensional data samples and complex data acquisition protocols. While the aforementioned shortcomings to PCA have been recognized individually in prior works, there has been a lack of holistic update to the principles of PCA that prominently addresses them in an integrated fashion. In this tutorial, we have aimed to address this gap through a GSP perspective to PCA, which stems from the graphical interpretation of a covariance matrix. Specifically, a covariance matrix admits a graphical interpretation with features of the given dataset as the nodes of the graph and the pairwise covariances informing the edge structure.

Our starting point was the direct theoretical equivalence between a PCA-based learning algorithm that assigns weights to individual principal components and a graph filter implemented on a covariance matrix as the graph. This equivalence demonstrated an alternative implementation of a PCA-based learning model in terms of a polynomial over the covariance matrix, which offers several key benefits over the

conventional PCA-based learning models: (a) graph filter implementation of PCA does not require an explicit eigendecomposition of the covariance matrix; (b) graph filters form a task-specific and parametric learning model unlike PCA, which is commonly adopted as a pre-processing step and decoupled from the learning task; and (c) scale-free characteristic implementation of graph filters allow them to be amenable to processing multiscale information with the same model, unlike PCA which is tied to the dimensionality of the given dataset. The expressive power of a graph filter could be enhanced by adding pointwise non-linearities and building banks of graph filters, leading towards VNN being defined as the GNN with a covariance matrix as the graph.

The surveyed theoretical results in this article rigorously demonstrated the robustness of VNN outcomes to stochastic perturbations in the covariance matrix as well as transference across datasets of different scales. Robustness of learning outcomes to stochastic perturbations in the covariance matrix enforces their reproducibility in finite-data regimes. This is because VNNs operate over the sample covariance matrix in practice, and the said robustness guarantees consistent learning outcomes across different sample covariance matrices from other finite-sized datasets sampled from the same distribution. Theoretical guarantees on the transference across multiscale datasets imply the effectiveness of VNNs at leveraging the redundancy of signals at different resolutions, which can potentially lead to computation- or data-efficient ML frameworks. While these theoretical guarantees reinforced the categorical advantages brought in by the GSP perspective to PCA, our aim here was not to diminish the PCA transform or put VNNs in contrast to it, but rather to offer significant conceptual advancements to PCA through VNN models. Achieving this goal involved bridging PCA with graph filters by demonstrating natural analytical and conceptual connections.

GSP-driven learning architectures have previously been shown to be robust to abstract perturbations in the graph as well as transferable across graphs of different sizes. While these properties extend naturally to VNNs, this tutorial article has also highlighted the unique insights brought in by the data-driven construction of the covariance matrix and the perturbation theory governing the divergence between the sample covariance matrix and its true counterpart. Specifically, we discussed the refined theoretical bounds in the context of VNNs, which enabled the understanding of their dependence on the data size and specific properties of the covariance matrix. GNNs are widely implemented on covariance matrices, and the theoretical results surveyed here contribute to their principled deployments in regimes with finite data. Furthermore, the discussions on various extensions of the VNNs to settings with fairness considerations, temporally evolving datasets, and corrupted data highlight the broad relevance of the theoretical principles surveyed herein to applications where covariance matrices emerge and PCA has been a workhorse analytical tool.

Looking ahead, this article has set the groundwork for enhancing the theoretical concepts of other multivariate statistical inference methods besides PCA as well as modern deep learning models that leverage covariances. For instance, a recent study of graph filters from the lens of neural tangent kernels [52] has revealed cross-covariance between the inputs and the desired outputs to contribute to the optimal generalization and training performance of a GNN [53]. This observation sets up a potential fundamental connection between a GNN with cross-covariance graphs and multivariate statistical methods, such as canonical correlation analysis (CCA), that explicitly leverage cross-covariance. In this context, future studies that address the shortcomings of conventional statistical methods (for instance, related to instability of CCA [54]) are well-motivated. Moreover, covariances are also a significant part of various prevalent deep learning architectures, such as transformers. The conceptual similarities between the self-attention mechanism and coVariance filters (highlighted in 'Self-Attention in Transformers versus CoVariance Filter in VNNs') reveal a promising direction towards theoretical understanding of transformers.

Furthermore, the foundational advances surveyed in this article can promisingly offer benefits in science and engineering applications where covariance matrices emerge. A recent tutorial article highlighted the key points of clarity and advancements offered by VNNs to the brain age gap prediction in the computational neuroscience domain [48]. Specifically, the VNN-driven brain age gap prediction pipeline offered a promising alternative to the currently prevalent ML approaches in this application that are largely opaque and performance-driven; and hence, not practical for clinical deployment. VNNs offer key benefits in this context due to enhanced interpretability and guaranteed stability and generalizability of anatomical patterns characterizing neurodegeneration. Furthermore, multiscale information is increasingly prevalent across various applications, For instance, the physical processes in atmospheric or oceanographic sciences are typically monitored with surface-level, airborne, and satellite-level sensors [55]; thus, revealing the need for learning models that can process multiscale information. Multiscale information is appealing in ML because it encodes the redundancy of information at different scales (spatial or temporal), potentially leading to computation- and sample-efficient training mechanisms and deployment via transference from small- to large-scale datasets. While recent works have proposed various deep learning algorithms that can succeed in inference tasks across multiscale datasets in specific domains, the mathematical principles behind such models are lacking and could be enhanced with the foundational advancements behind VNNs.

In summary, VNNs hold great promise to drive the necessary conceptual updates to conventional statistical methods to meet the needs of modern data analysis and enhance the theoretical understanding of modern deep learning methods. We argue that VNNs are particularly useful in finite-data regimes, where their theoretical guarantees ensure consistent and reliable learning outcomes. Furthermore, the theory of VNNs can lend unparalleled depth while offering major conceptual advancements in the design

of application-relevant data analysis solutions.

ACKNOWLEDGMENT

CamCAN dataset was obtained from the CamCAN repository [44], [56]. Dallas Lifespan Brain Study (DLBS) is available at International Neuroimaging Data-sharing Initiative (INDI). IXI dataset is available at https://brain-development.org/ixi-dataset/. The eNKI dataset is available at https://fcon_1000.projects.nitrc.org/indi/pro/eNKI_RS_TRT/FrontPage.html.

REFERENCES

[1] A. C. Evans, "Networks of anatomical covariance," *Neuroimage*, vol. 80, pp. 489–504, 2013.

[2] H. Murase and S. K. Nayar, "Illumination planning for object recognition using parametric eigenspaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 12, pp. 1219–1227, 1994.

[3] D. Stephenson, "Correlation of spatial climate/weather maps and the advantages of using the mahalanobis metric in predictions," *Tellus A*, vol. 49, no. 5, pp. 513–527, 1997.

[4] H. Shao, W. H. Lam, A. Sumalee, A. Chen, and M. L. Hazelton, "Estimation of mean and covariance of peak hour origin–destination demands from day-to-day traffic counts," *Transport. Res. B-Meth.*, vol. 68, pp. 52–75, 2014.

[5] M. N. Ismail, A. Aborujilah, S. Musa, and A. Shahzad, "Detecting flooding based DoS attack in cloud computing environment using covariance matrix approach," in *Proc. Int. Conf. Ubiquitous Inf. Manag. Commun.*, 2013, pp. 1–6.

[6] M. Greenacre, P. Groenen, T. Hastie, A. Iodice D'Enza, A. Markos, and E. Tuzhilina, "Principal component analysis," *Nature Reviews Methods Primers*, vol. 2, p. 100, 12 2022.

[7] J. Shlens, "A tutorial on principal component analysis," *arXiv preprint arXiv:1404.1100*, 2014.

[8] I. T. Joliffe and B. Morgan, "Principal component analysis and exploratory factor analysis," *Stat. Methods Med. Res.*, vol. 1, no. 1, pp. 69–95, 1992.

[9] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.

[10] A. Loukas, "How close are the eigenvectors of the sample and actual covariance matrices?" in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2228–2237.

[11] A. S. Willsky, "Multiresolution Markov models for signal and image processing," *Proc. IEEE*, vol. 90, no. 8, pp. 1396–1458, 2002.

[12] G. Leus, A. G. Marques, J. M. Moura, A. Ortega, and D. I. Shuman, "Graph signal processing: History, development, impact, and outlook," *IEEE Signal Process. Mag.*, vol. 40, no. 4, pp. 49–60, 2023.

[13] L. Ruiz, F. Gama, and A. Ribeiro, "Graph neural networks: architectures, stability, and transferability," *Proc. IEEE*, vol. 109, no. 5, pp. 660–682, 2021.

[14] F. Gama, E. Isufi, G. Leus, and A. Ribeiro, "Graphs, convolutions, and neural networks: From graph filters to graph neural networks," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 128–138, 2020.

[15] F. Gama, J. Bruna, and A. Ribeiro, "Stability properties of graph neural networks," *IEEE Trans. Signal Process.*, vol. 68, pp. 5680–5695, 2020.

[16] R. Li, X. Yuan, M. Radfar, P. Marendy, W. Ni, T. J. O'Brien, and P. M. Casillas-Espinosa, "Graph signal processing, graph neural network and graph learning on biological data: a systematic review," *IEEE Reviews in Biomedical Engineering*, vol. 16, pp. 109–135, 2021.

[17] E. Isufi, F. Gama, D. I. Shuman, and S. Segarra, "Graph filters for signal processing and machine learning on graphs," *IEEE Transactions on Signal Processing*, vol. 72, pp. 4745–4781, 2024.

[18] S. Sihag, G. Mateos, C. McMillan, and A. Ribeiro, "coVariance neural networks," in *Proc. Conf. Adv. in Neural Inf. Process. Syst.*, Nov. 2022.

[19] A. Cavallo, Z. Gao, and E. Isufi, "Sparse covariance neural networks," *arXiv:2410.01669*, vol. cs.LG, 2024. [Online]. Available: https://arxiv.org/abs/2410.01669

[20] A. Cavallo, M. Sabbaqi, and E. Isufi, "Spatiotemporal covariance neural networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2024, pp. 18–34.

[21] A. Cavallo, M. Navarro, S. Segarra, and E. Isufi, "Fair covariance neural networks," *arXiv preprint arXiv:2409.08558*, 2024.

[22] C. Davis and W. M. Kahan, "The rotation of eigenvectors by a perturbation. iii," *SIAM Journal on Numerical Analysis*, vol. 7, no. 1, pp. 1–46, 1970.

[23] T. Szwagier and X. Pennec, "The curse of isotropy: from principal components to principal subspaces," *arXiv preprint arXiv:2307.15348*, 2023.

[24] J. Janková and S. van de Geer, "De-biased sparse pca: Inference for eigenstructure of large covariance matrices," *IEEE Transactions on Information Theory*, vol. 67, no. 4, pp. 2507–2527, 2021.

[25] Z. T. Ke, L. Xue, and F. Yang, "Diagonally dominant principal component analysis," *Journal of Computational and Graphical Statistics*, vol. 29, no. 3, pp. 592–607, 2020.

[26] Y. Xie, T. Wang, J. Kim, K. Lee, and M. K. Jeong, "Least angle sparse principal component analysis for ultrahigh dimensional data," *Annals of Operations Research*, pp. 1–27, 2024.

[27] A. Del Pia, "Sparse pca on fixed-rank matrices," *Mathematical Programming*, vol. 198, no. 1, pp. 139–157, 2023.

[28] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2019, pp. 2623–2631.

[29] S. Sihag, G. Mateos, C. McMillan, and A. Ribeiro, "Transferability of covariance neural networks," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–16, 2024.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT press, 2016.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[32] C. K. Joshi, "Transformers are graph neural networks," *arXiv preprint arXiv:2506.22084*, 2025.

[33] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do vision transformers see like convolutional neural networks?" *Advances in neural information processing systems*, vol. 34, pp. 12 116–12 128, 2021.

[34] Y. Deshpande and A. Montanari, "Sparse pca via covariance thresholding," *Journal of Machine Learning Research*, vol. 17, no. 141, pp. 1–41, 2016.

[35] P. J. Bickel and E. Levina, "Covariance regularization by thresholding," *The Annals of Statistics*, pp. 2577–2604, 2008.

[36] C. Wang, R. Tian, J. Hu, and Z. Ma, "A trend graph attention network for traffic prediction," *Information Sciences*, vol. 623, pp. 275–292, 2023.

[37] J. Wang, L. Lin, Z. Zhang, S. Gao, and H. Yu, "Deep neural network based on dynamic attention and layer attention for meteorological data downscaling," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 215, pp. 157–176, 2024.

[38] A. Schaefer, R. Kong, E. M. Gordon, T. O. Laumann, X.-N. Zuo, A. J. Holmes, S. B. Eickhoff, and B. T. Yeo, "Local-global parcellation of the human cerebral cortex from intrinsic functional connectivity MRI," *Cerebral Cortex*, vol. 28, no. 9, pp. 3095–3114, 2018.

[39] N. Cressie and J. Kornak, "Spatial statistics in the presence of location error with an application to remote sensing of the environment," *Statistical science*, pp. 436–456, 2003.

[40] L. Ruiz, L. F. Chamon, and A. Ribeiro, "Graphon signal processing," *IEEE Trans. Signal Process.*, vol. 69, pp. 4961–4976, 2021.

[41] C. Borgs, J. T. Chayes, L. Lovász, V. T. Sós, and K. Vesztergombi, "Convergent sequences of dense graphs i: Subgraph frequencies, metric properties and testing," *Adv. Math.*, vol. 219, no. 6, pp. 1801–1851, 2008.

[42] L. Lovász, *Large Networks and Graph Limits*. American Mathematical Soc., 2012, vol. 60.

[43] I. Gohberg, S. Goldberg, M. A. Kaashoek, I. Gohberg, S. Goldberg, and M. A. Kaashoek, "Hilbert-schmidt operators," *Classes of Linear Operators Vol. I*, pp. 138–147, 1990.

[44] M. A. Shafto, L. K. Tyler, M. Dixon, J. R. Taylor, J. B. Rowe, R. Cusack, A. J. Calder, W. D. Marslen-Wilson, J. Duncan, T. Dalgleish *et al.*, "The cambridge centre for ageing and neuroscience (cam-can) study protocol: a cross-sectional, lifespan, multidisciplinary examination of healthy cognitive ageing," *BMC neurology*, vol. 14, pp. 1–25, 2014.

[45] K. B. Nooner, S. J. Colcombe, R. H. Tobe, M. Mennes, M. M. Benedict, A. L. Moreno, L. J. Panek, S. Brown, S. T. Zavitz, Q. Li *et al.*, "The nki-rockland sample: a model for accelerating the pace of discovery science in psychiatry," *Frontiers in neuroscience*, vol. 6, p. 152, 2012.

[46] C. Gaser, R. Dahnke, P. M. Thompson, F. Kurth, E. Luders, and Alzheimer's Disease Neuroimaging Initiative, "CAT–a computational anatomy toolbox for the analysis of structural mri data," *biorxiv*, pp. 2022–06, 2022.

[47] S. Sihag, G. Mateos, C. McMillan, and A. Ribeiro, "Explainable brain age prediction using covariance neural networks," in *Proc. Conf. Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=cAhJF87GN0

[48] S. Sihag, G. Mateos, and A. Ribeiro, "Disentangling neurodegeneration with brain age gap prediction models: A graph signal processing perspective," *IEEE Signal Processing Magazine*, vol. 42, no. 4, pp. 58–77, 2025.

[49] A. Cavallo, A. Raghuvanshi, S. P. Chepuri, and E. Isufi, "Robust covariance neural networks," in *2025 59th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2025.

[50] ——, "Covariance scattering transforms," *arXiv preprint arXiv:2511.08878*, 2025.

[51] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. of the ACM (JACM)*, vol. 58, no. 3, pp. 1–37, 2011.

[52] A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[53] S. Khalafi, S. Sihag, and A. Ribeiro, "Neural tangent kernels motivate cross-covariance graphs in neural networks," in *Forty-first International Conference on Machine Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=61JD8wp4Id

[54] M. Helmer, S. Warrington, A.-R. Mohammadi-Nejad, J. L. Ji, A. Howell, B. Rosand, A. Anticevic, S. N. Sotiropoulos, and J. D. Murray, "On the stability of canonical correlation analysis and partial least squares with application to brain-behavior associations," *Communications biology*, vol. 7, no. 1, p. 217, 2024.

[55] M. Neelam, A. Colliander, B. P. Mohanty, M. H. Cosh, S. Misra, and T. J. Jackson, "Multiscale surface roughness for improved soil moisture estimation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 8, pp. 5264–5276, 2020.

[56] J. R. Taylor, N. Williams, R. Cusack, T. Auer, M. A. Shafto, M. Dixon, L. K. Tyler, R. N. Henson *et al.*, "The cambridge centre for ageing and neuroscience (cam-can) data repository: Structural and functional mri, meg, and cognitive data from a cross-sectional adult lifespan sample," *neuroimage*, vol. 144, pp. 262–269, 2017.

BIOGRAPHIES

Saurabh Sihag is an Assistant Professor in the Department of Electrical and Computer Engineering at the University at Albany. He received his PhD degree in Electrical Engineering from Rensselaer Polytechnic Institute, in 2020, and his Bachelor's and Master's degrees in Electrical Engineering from Indian Institute of Technology, Kharagpur in 2016. He was a postdoctoral researcher in the Department of Electrical and Systems Engineering at the University of Pennsylvania between March, 2021 and July, 2024. He has previously been the recipient of J. Baliga fellowship and Charles M. Close '62 Doctoral Prize for his doctoral dissertation. His research interests include statistical inference and machine learning over graph-structured data and network neuroscience.

Andrea Cavallo received his B.Sc. and M.Sc. degrees (cum laude) in Computer Engineering from the Polytechnic University of Turin, Italy, in 2020 and 2022, respectively. Since 2023, he is a Ph.D. candidate at the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Netherlands. His research focuses on relational machine learning through higher-order networks and statistical moments as well as temporal network modeling and prediction.

Elvin Isufi is an Associate Professor in the Faculty of Electrical Engineering, Mathematics and Computer Science at Delft University of Technology (TU Delft), The Netherlands, where he also co-directs AIdroLab –the TU Delft AI Lab on graph-based learning for water networks. He received the Ph.D. degree from the same university ('19), and the M. Sc. ('14) and B. Sc. ('12) in Electronic and Telecommunication Engineering from the University of Perugia, Italy. His research interests are in signal processing and machine learning for graphs and other topological structures with applications in critical infrastructure networks, recommender systems, and sensor networks. He received the 2022 IEEE SPS Best PhD Dissertation Award, the 2025 ICASSP Best Conference Paper Award and paper recognition awards at the IEEE CAMSAP ('17), DSLW ('21, '22), and ICASSP ('23). He is a member of the IEEE SPS Technical Committee on Signal Processing for Communication and Networking and serves as Associate Editor for the IEEE Transactions on Signal Processing, position that he also had held for Elsevier Signal Processing ('24-'26). He is an NWO VENI fellow ('24) and a Horizon MSCA Cofound fellow ('19).

Gonzalo Mateos received his B.Sc. degree in Electrical Engineering from Universidad de la República, Montevideo, Uruguay in 2005 and the M.Sc. and Ph.D. degrees in Electrical Engineering from the University of Minnesota, Minneapolis, in 2009 and 2012. Currently, he is a Professor with the Department of Electrical and Computer Engineering, University of Rochester, as well as the Associate Director for Research at the Goergen Institute for Data Science and Artificial Intelligence. He also was an Asaro Biggar Family Fellow in Data Science (2020-23). His research interests lie in the areas of statistical learning from complex data, network science, decentralized optimization, and graph signal processing.

Alejandro Ribeiro received the B.Sc. degree in Electrical Engineering from the Universidad de la República, Montevideo, Uruguay, in 1998, the M.Sc. and Ph.D. degrees in Electrical Engineering from the University of Minnesota, Minneapolis, MN, USA, in 2005 and 2007, respectively. Since 2008, he has been with the University of Pennsylvania (Penn), Philadelphia, PA, USA, where he is currently a Professor of Electrical and Systems Engineering. His research interests include applications of statistical signal processing to the study of networks and networked phenomena, structured representations of networked data structures, graph signal processing, network optimization, robot teams, and networked control. Dr. Ribeiro was the recipient of an Outstanding Research Award from Intel in 2019, the 2014 O. Hugo Schuck Best Paper Award, and paper awards at ICASSP 2020, EUSIPCO 2019, CDC 2017, 2016 SSP Workshop, 2016 SAM Workshop, 2015 Asilomar SSC, ACC 2013, ICASSP 2006, and ICASSP 2005. His teaching has been recognized with the 2017 Lindback Award and the 2012 S. Reid Warren, Jr. Award presented by Penn's undergraduate student body for outstanding teaching. Dr. Ribeiro is a Fulbright Scholar class of 2003 and a Penn Fellow class of 2015.