TU Delft

Delft University of Technology

Document Version
Final published version

Licence
Dutch Copyright Act (Article 25fa)

Citation (APA)
Cavallo, A., Rey, S., Marques, A. G., & Isufi, E. (2026). Precision Neural Networks: Joint Graph and Relational Learning. In *Proceedings of the ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 466-470). IEEE. https://doi.org/10.1109/ICASSP55912.2026.11463610

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3315-6701-9/26/$31.00 ©2026 IEEE | DOI: 10.1109/ICASSP55912.2026.11463610

PRECISION NEURAL NETWORKS: JOINT GRAPH AND RELATIONAL LEARNING

Andrea Cavallo[*], *Samuel Rey*[†], *Antonio G. Marques*[†], *Elvin Isufi*[*]

[*]Delft University of Technology, Delft, Netherlands
[†]King Juan Carlos University, Madrid, Spain

ABSTRACT

CoVariance Neural Networks (VNNs) perform convolutions on the graph determined by the covariance matrix of the data, which enables expressive and stable covariance-based learning. However, covariance matrices are typically dense, fail to encode conditional independence, and are often precomputed in a task-agnostic way, which may hinder performance. To overcome these limitations, we study *Precision Neural Networks (PNNs)*, i.e., VNNs on the precision matrix—the inverse covariance. The precision matrix naturally encodes statistical independence, often exhibits sparsity, and preserves the covariance spectral structure. To make precision estimation task-aware, we formulate an optimization problem that *jointly learns the network parameters and the precision matrix*, and solve it via alternating optimization, by sequentially updating the network weights and the precision estimate. We theoretically bound the distance between the estimated and true precision matrices at each iteration, and demonstrate the effectiveness of joint estimation compared to two-step approaches on synthetic and real-world data.

***Index Terms*—** Precision Neural Networks, Graphical Lasso, Graph Neural Networks

1. INTRODUCTION

Capturing relations among data points in complex systems is fundamental to their effective modeling and understanding. In applications such as brain data [1, 2], sensor measurements [3], and finance [4, 5], data interactions are commonly captured via covariance-based processing. A classical approach is Principal Component Analysis (PCA) [6], which projects the data onto the eigenvectors of the covariance matrix to maximize variance and, when desired, reduce dimensionality. While effective as a preprocessing step before applying a machine learning (ML) model for a downstream task (cf. Figure 1), PCA is sensitive to noise and estimation errors, and lacks flexibility in processing the covariance eigenvectors. To overcome these limitations, coVariance Neural Networks (VNNs) [7] have emerged as powerful tools thanks to their stability to finite sample noise, expressivity, and ability to learn directly from data correlations. VNNs are sequences of nonlinearities interleaved with banks of covariance filters, i.e., polynomials of the covariance matrix, which can be interpreted as convolutions on the graph determined by the covariance matrix. This enables VNNs to learn expressive functions in the covariance eigenspace, thus extending PCA towards covariance-aware learning for downstream tasks (cf. Figure 1), while being stable to covariance estimation errors. This makes VNNs robust in low-data regimes, favorable for interpretability [8] and transferable across resolutions [9], and these benefits extend to spatiotemporal data [10] and to covariance estimators promoting sparsity [11] and fairness [12]. However, VNNs and their extensions still rely on precomputed covariances, which decouples correlation estimation from the downstream objective (e.g., brain connectivity in healthy versus diseased subjects, or horizon-specific financial dependencies). Moreover, the covariance matrix only reflects marginal correlations and cannot distinguish direct from indirect effects, making it insufficient for capturing conditional independence and often leading to redundant dependency structures.

As an alternative, we study the application of VNNs to the precision matrix, thus defining *Precision Neural Networks (PNNs)*, and we propose a joint estimation of the precision and the neural network (NN) weights. The precision matrix is typically sparse and encodes conditional independence among variables, thereby capturing covariance information in a more efficient and sparse representation and more effectively modeling data relations [13]. Moreover, it shares the eigenvectors of the covariance matrix, which enables PNNs to perform the same spectral task as VNNs. However, directly inverting the sample covariance is unreliable in high-dimensional regimes, and existing sparse precision estimators such as graphical lasso [14–16] are task-agnostic. To overcome this limitation, we introduce a framework that jointly learns both the sparse precision matrix and the weights of the PNN, yielding a *task-aware precision* that balances data fidelity with downstream performance and can identify connectivity patterns that are relevant to solving the task at hand. Since the resulting problem is challenging due to the multiple optimization variables, we address it through an alternating strategy that sequentially updates the network parameters and the precision. We provide theoretical insights by proving that, at each iteration, the estimated precision converges to the true matrix at a rate inversely proportional to the square root of the number of observations. While several works analyze joint estimation of a graph structure and a model to process it [17–21], they either focus on graph filters only, they lack a theoretical analysis of the estimated graph or they do not build a connection with covariance-based learning methods.

We summarize our main contributions below.

(C1) We study PNNs, graph convolutional NNs on sparse precision matrices that leverage conditional independence through an efficient representation.

(C2) We develop a joint estimation framework for the precision matrix and the PNN weights, yielding a task-aware precision matrix that adapts to the downstream objective.

(C3) We provide theoretical insights on the quality of the estimated precision at each iteration, and demonstrate that PNNs outperform competitive methods on synthetic data and on an age prediction task from cortical thickness measures.

Work partially funded by the TU Delft AI Labs programme, the NWO OTP GraSPA proposal #19497, the NWO VENI proposal 222.032, the Spanish AEI (10.13039/501100011033) grant PID2022-136887NB-I00, and the Community of Madrid via the Ellis Madrid Unit and grants URJC/CAM F1180 and TEC-2024/COM-89. Emails: {a.cavallo, e.isufi-1}@tudelft.nl, {samuel.rey.escudero, antonio.garcia.marques}@urjc.es.

Authorized licensed use limited to: TU Delft Library. Downloaded on May 12,2026 at 10:23:32 UTC from IEEE Xplore. Restrictions apply.

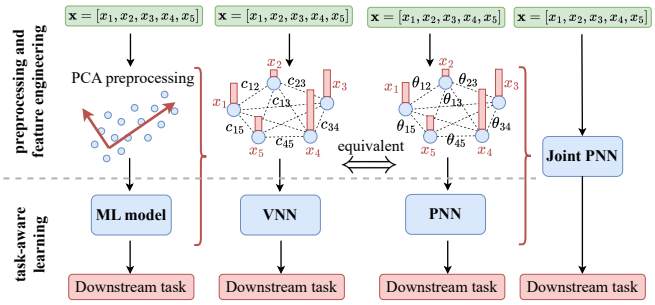


Fig. 1: Data processing via covariance information. Traditional pipelines use PCA for preprocessing followed by an ML model for the downstream task. VNNs and PNNs merge these steps by learning end-to-end on the covariance or the precision graph. Joint PNNs unify the estimation of the precision matrix and the network weights.

2. PRECISION NEURAL NETWORKS

We define PNNs, graph convolutional NNs operating on the inverse covariance, known as the precision matrix.

Preliminaries. Consider a data matrix $\mathbf{X} \in \mathbb{R}^{N\times T}$ (zero-mean w.l.o.g.) whose columns $\mathbf{x}_i \in \mathbb{R}^N$ are observations of a random variable $\mathbf{x}$ with covariance $\mathbf{C}_0 = \mathbb{E}[(\mathbf{x}-\mathbb{E}[\mathbf{x}])(\mathbf{x}-\mathbb{E}[\mathbf{x}])^\mathsf{T}]$. Since $\mathbf{C}_0$ is generally unknown, it is approximated by the sample estimate $\mathbf{C} = \mathbf{X}\mathbf{X}^\mathsf{T}/T$ with eigendecomposition $\mathbf{C} = \mathbf{V}\mathbf{W}\mathbf{V}^\mathsf{T}$, where the diagonal matrix $\mathbf{W} \in \mathbb{R}^{N\times N}$ collects the eigenvalues and $\mathbf{V} \in \mathbb{R}^{N\times N}$ contains the orthogonal eigenvectors. The precision matrix and its estimate are, respectively, $\mathbf{\Theta}_0 = \mathbf{C}^{-1}$ and $\mathbf{\Theta}$.

PNN definition. A PNN consists of L layers, each comprising a (graph-based) filter bank of size $F_{\text{in}}^{(\ell)} \times F_{\text{out}}^{(\ell)}$ followed by a nonlinear activation function σ. For layer $\ell = 1,\dots,L$ and output feature $f = 1,\dots,F_{\text{out}}^{(\ell)}$, the precision filter and propagation rule are

$$\mathbf{x}_f^{(\ell)} = \sigma\left(\sum_{j=1}^{F_{\text{in}}} \mathbf{H}_{fj}^{(\ell)}(\mathbf{\Theta})\mathbf{x}_j^{(\ell-1)}\right), \tag{1}$$

$$\mathbf{H}_{fj}^{(\ell)}(\mathbf{\Theta}) = \sum_{k=0}^{K} h_{kfj}^{(\ell)}\mathbf{\Theta}^k. \tag{2}$$

The PNN output $\mathbf{x}^{(L)}$ is passed to a readout function, typically a Multi-Layer Perceptron (MLP), to solve the downstream task. We denote the overall pipeline, comprising the PNN and the readout, as $\Phi(\mathbf{x},\mathbf{\Theta},\mathbf{h})$, where $\mathbf{x}$ is an input signal and $\mathbf{h}$ is a vector stacking all polynomial coefficients $h_{kfj}^{(\ell)}$ in the PNN. Given a training set $(\mathbf{X}_{\text{tr}},\mathbf{y}_{\text{tr}})$, where $\mathbf{X}_{\text{tr}} \in \mathbb{R}^{N\times T}$ contains the input signals and $\mathbf{y}_{\text{tr}} \in \mathbb{R}^T$ the corresponding labels or regression targets, the weights $\mathbf{h}$ are learned by solving

$$\min_{\mathbf{h}} \mathcal{L}_{\text{task}}(\mathbf{y}_{\text{tr}},\Phi(\mathbf{X}_{\text{tr}},\mathbf{\Theta},\mathbf{h})) + \beta\|\mathbf{h}\|_2^2 \tag{3}$$

where $\mathcal{L}_{\text{task}}$ is a task-specific loss (e.g., cross-entropy for classification or MSE for regression), and $\beta \geq 0$ is a regularization parameter.

The use of precision matrices in a deep NN enjoys three key benefits. First, while the covariance matrix encodes pairwise correlations and is typically dense, $\mathbf{\Theta}$ often admits a sparse structure, enabling more efficient computations. Second, when the observed data $\mathbf{x}$ follows a Gaussian distribution, $\mathbf{\Theta}$ encodes conditional independence, with $[\mathbf{\Theta}]_{ij} = 0$ if and only if $[\mathbf{x}]_i$ and $[\mathbf{x}]_j$ are conditionally independent given the rest [14]. Finally, $\mathbf{\Theta}$ is closely connected to Gaussian graphical models, so that (2) can be interpreted as a convolution on the graph induced by the precision matrix.

Connection to PCA. An important property of $\mathbf{\Theta}$ is that it shares the eigenvectors of $\mathbf{C}$, so that $\mathbf{\Theta} = \mathbf{V}\mathbf{M}\mathbf{V}^\mathsf{T}$ with $\mathbf{M} = \mathbf{W}^{-1}$. Following the graph signal processing literature [22], we define the precision Fourier transform as the projection of a signal $\mathbf{x}$ on the eigenvectors of the precision matrix, i.e., $\tilde{\mathbf{x}} = \mathbf{V}^\mathsf{T}\mathbf{x}$. Since $\mathbf{V}$ is also the eigenbasis of $\mathbf{C}$, this transform coincides with the PCA projection. Moreover, in the spectral domain, PNNs learn polynomial functions of the eigenvalues of the precision matrix $\{\mu_i\}_{i=1}^N$. In detail, the action of a single precision filter on the i-th component of the projected signal $\tilde{\mathbf{x}}$ is

$$[\tilde{\mathbf{x}}]_i^{(\ell)} = \sum_{k=0}^{K} h_k^{(\ell)}\mu_i^k[\tilde{\mathbf{x}}]_i^{(\ell-1)} = h^{(\ell)}(\mu_i)[\tilde{\mathbf{x}}]_i^{(\ell-1)}. \tag{4}$$

Thus, PNNs modulate the data projections on their principal components $[\tilde{\mathbf{x}}]_i$ via their frequency response $h(\mu)$.

Connection to VNNs. VNNs are graph convolutional NNs using the sample covariance as shift operator—i.e., they replace $\mathbf{\Theta}$ with $\mathbf{C}$ in (1) and thus they learn a polynomial frequency response analogous to (4) on the eigenvalues $w = \mu^{-1}$. Consequently, if the filter degrees K are sufficiently large, VNNs and PNNs can learn the same transformation in the spectral domain by properly adjusting the coefficients $h_k^{(\ell)}$.

3. JOINT ESTIMATION OF PNN PARAMETERS AND PRECISION MATRIX

PNNs require an estimate of the precision matrix to be available. A simple approach is to compute $\mathbf{\Theta}$ by inverting the sample covariance $\mathbf{C}$, but this is known to yield poor estimates, particularly in high-dimensional or low-sample regimes [15]. Alternatively, one may employ established methods such as graphical Lasso, which directly estimate $\mathbf{\Theta}$ under structural assumptions like sparsity. Although these approaches typically provide more reliable precision matrices, they remain task-agnostic, since the estimation procedure does not incorporate the downstream objective. As a result, the obtained precision matrix may still be suboptimal for the learning problem at hand. To make the estimation fully task-aware, we consider a *joint estimation of the PNN weights and the precision matrix*. Given a training set $(\mathbf{X}_{\text{tr}},\mathbf{Y}_{\text{tr}})$, we minimize

$$\begin{aligned}\min_{\mathbf{h},\mathbf{\Theta}}\ & \alpha\mathcal{L}_{\text{task}}(\mathbf{y}_{\text{tr}},\Phi(\mathbf{X}_{\text{tr}},\mathbf{\Theta},\mathbf{h})) + (1-\alpha)\mathcal{L}_{\text{GL}}(\mathbf{X}_{\text{tr}},\mathbf{\Theta}) \\ \text{s.t.}\ & \mathbf{\Theta} \succeq 0, \quad \|\mathbf{\Theta}\|_2 \leq M,\end{aligned} \tag{5}$$

where $\alpha \in [0,1]$, $\mathcal{L}_{\text{task}}$ is the task-specific loss from (3), and $\mathcal{L}_{\text{GL}}$ is the graphical Lasso objective [15], given by

$$\mathcal{L}_{\text{GL}}(\mathbf{X}_{\text{tr}},\mathbf{\Theta}) = \text{tr}(\mathbf{C}\mathbf{\Theta}) - \text{logdet}(\mathbf{\Theta}+\epsilon\mathbf{I}) + \lambda\|\mathbf{\Theta}_{\bar{\mathcal{D}}}\|_1. \tag{6}$$

Here, $\mathbf{\Theta}_{\bar{\mathcal{D}}}$ denotes the off-diagonal elements of $\mathbf{\Theta}$, the ℓ_1 norm promotes sparsity, while the regularization $\epsilon\mathbf{I}$ with $\epsilon \geq 0$ enables learning positive semi-definite precision matrices. $\mathbf{\Theta}_0$ is assumed to have bounded eigenvalues (see assumptions in Theorem 1), hence the constraint $\|\mathbf{\Theta}\|_2 \leq M$. In practice, an effective M can be obtained by overshooting its value based on the minimum eigenvalue of the sample covariance. The parameter α balances the influence of the task-driven and model-driven terms, yielding a precision matrix $\mathbf{\Theta}$ that supports the PNN in solving the downstream task. However, the joint optimization over $\mathbf{\Theta}$ and $\mathbf{h}$ poses a coupled and more challenging landscape, as it leads to complex and potentially unstable gradients [17].

Authorized licensed use limited to: TU Delft Library. Downloaded on May 12,2026 at 10:23:32 UTC from IEEE Xplore. Restrictions apply.

3.1. Relaxed Formulation and Optimization.

To partially alleviate these issues, we start by adopting the following relaxed formulation:

$$\min_{\mathbf{h},\tilde{\mathbf{\Theta}}} \quad \alpha\mathcal{L}_{\text{task}}(\mathbf{y}_{\text{tr}}, \Phi(\mathbf{X}_{\text{tr}}, \tilde{\mathbf{\Theta}}, \mathbf{h})) + (1-\alpha)\mathcal{L}_{\text{GL}}(\mathbf{X}_{\text{tr}}, \mathbf{\Theta}) + \frac{\gamma}{2}\|\mathbf{\Theta} - \tilde{\mathbf{\Theta}}\|_F^2 \quad \text{s.t.} \quad \mathbf{\Theta} \succeq 0, \quad \|\mathbf{\Theta}\|_2 \leq M, \tag{7}$$

where the auxiliary variable $\tilde{\mathbf{\Theta}}$ decouples $\mathcal{L}_{\text{task}}$ from $\mathcal{L}_{\text{GL}}$ through a quadratic penalty. Notably, in the limit of $\gamma \to \infty$, the constraint $\mathbf{\Theta} = \tilde{\mathbf{\Theta}}$ is enforced exactly, and problem (7) reduces to (5). This alternative formulation is amenable to an alternating optimization strategy [17, 23]. In particular, we approach it by sequentially solving the following steps for $i = 1, \ldots, I_{\text{epochs}}$ iterations.

Step 1. We update $\mathbf{\Theta}$ with $\tilde{\mathbf{\Theta}}^{(i)}$ fixed by solving

$$\mathbf{\Theta}^{(i+1)} = \underset{\mathbf{\Theta}}{\operatorname{argmin}} (1-\alpha)\mathcal{L}_{\text{GL}}(\mathbf{X}_{\text{tr}}, \mathbf{\Theta}) + \frac{\gamma}{2}\|\mathbf{\Theta} - \tilde{\mathbf{\Theta}}^{(i)}\|_F^2. \quad \text{s.t.} \quad \mathbf{\Theta} \succeq 0, \quad \|\mathbf{\Theta}\|_2 \leq M. \tag{8}$$

This step amounts to a convex optimization problem closely related to graphical Lasso, which can be solved efficiently with existing algorithms. We adopt a proximal gradient descent approach [24, 25] where gradient descent is applied to the smooth terms in (8), followed by the proximal operator of the ℓ_1 norm (soft-thresholding of the off-diagonal elements), and finally projection onto the cone of positive semidefinite matrices and norm clipping.

Step 2. Next, we update $\tilde{\mathbf{\Theta}}$ while fixing $\mathbf{h}^{(i)}$ and $\mathbf{\Theta}^{(i+1)}$. This requires solving

$$\tilde{\mathbf{\Theta}}^{(i+1)} = \underset{\tilde{\mathbf{\Theta}}}{\operatorname{argmin}} \alpha\mathcal{L}_{\text{task}}(\mathbf{y}_{\text{tr}}, \Phi(\mathbf{X}_{\text{tr}}, \tilde{\mathbf{\Theta}}, \mathbf{h}^{(i)})) + \frac{\gamma}{2}\|\mathbf{\Theta}^{(i+1)} - \tilde{\mathbf{\Theta}}\|_F^2. \tag{9}$$

In practice, this step is implemented via a forward pass through the PNN, a computation of the task-related loss and the Frobenius penalty, and gradient descent updates.

Step 3. Finally, we update the PNN weights while keeping the previous estimates $\mathbf{\Theta}^{(i+1)}$ and $\tilde{\mathbf{\Theta}}^{(i+1)}$ fixed. This amounts to solving

$$\mathbf{h}^{(i+1)} = \underset{\mathbf{h}}{\operatorname{argmin}} \alpha\mathcal{L}_{\text{task}}(\mathbf{y}_{\text{tr}}, \Phi(\mathbf{X}_{\text{tr}}, \tilde{\mathbf{\Theta}}^{(i+1)}, \mathbf{h})), \tag{10}$$

which corresponds to standard NN training and can be performed with, e.g., the Adam optimizer [26]. We summarize the overall procedure in Algorithm 1 where we let $\mathcal{L}(\mathbf{h}, \mathbf{\Theta}, \tilde{\mathbf{\Theta}})$ denote the objective funtion in (7). Next, we take advantage of the convexity of Step 1 to characterize the quality of the estimated precision matrix.

3.2. Theoretical Analysis

We analyze the error between the sparse precision matrix $\mathbf{\Theta}^{(i)}$ estimated at iteration i and the true precision matrix $\mathbf{\Theta}_0$, as a function of the number of samples T and the dimension N. Our main result is stated in the following theorem.

Theorem 1 *Let $\mathbf{\Theta}^{(i)}$ denote the minimizer of the objective in* (8) *with $\lambda \asymp \sqrt{\log N/T}$ at iteration i, and let $\mathbf{\Theta}_0$ be the true precision matrix. Denote by S the number of nonzero off-diagonal entries of $\mathbf{\Theta}_0$, set $\epsilon = 0$, assume that the eigenvalues of $\mathbf{C}_0$ lie in $[k_{\min}, k_{\max}]$ for some constants $k_{\min} > 0$ and $k_{\max} < \infty$, and assume observations $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_0)$ i.i.d.. Then, with probability tending to one as $N, T \to \infty$, there exist constants $m_1, m_2 > 0$ such that*

$$\|\mathbf{\Theta}^{(i)} - \mathbf{\Theta}_0\|_F \leq m_1\sqrt{\frac{(N+S)\log N}{T}} + m_2\|\mathbf{\Theta}_0 - \tilde{\mathbf{\Theta}}^{(i)}\|_*^{1/2}. \tag{11}$$

where $\|\cdot\|_*$ denotes the nuclear norm.

Algorithm 1 Iterative optimization of $\mathbf{\Theta}$, $\tilde{\mathbf{\Theta}}$, and $\mathbf{h}$

```
Initialize Θ^(1), Θ̃^(1), h^(1)
for epoch = 1, ..., I_epochs do
    for j = 1, ..., I_Θ do
        Θ^(i+1) ← Θ^(i) − η ( C − (Θ^(i) + εI)^{-1} + γ(Θ^(i) − Θ̃^(i)) )
        Θ^(i+1) ← soft_threshold(Θ^(i+1), ηλ)
        Θ^(i+1) ← psd_projection(Θ^(i+1))
        Θ^(i+1) ← M / max(M, ||Θ^(i+1)||_2) Θ^(i+1)
    end
    for k = 1, ..., I_Θ̃ do
        Θ̃^(i+1) ← Θ̃^(i) − η∇_Θ̃ L(h^(i), Θ^(i+1), Θ̃^(i))
    end
    for l = 1, ..., I_h do
        h^(i+1) ← adam_update(h^(i), L(h^(i), Θ^(i+1), Θ̃^(i+1)))
    end
end
Return Θ^(I_epochs I_Θ + 1), Θ̃^(I_epochs I_Θ̃ + 1), h^(I_epochs I_h + 1)
```

Proof. The proof is omitted due to space limitations, we refer interested readers to the Appendix of [27].

Theorem 1 shows that the sequence of estimates $\mathbf{\Theta}^{(i)}$ obtained from solving (8) converges to the true precision $\mathbf{\Theta}_0$ as the number of samples increases, with rate $\mathcal{O}(T^{-1/2})$. The error bound also depends on the square root of the number of nonzero entries S, which makes the method particularly effective when the true precision is highly sparse ($S \ll N^2$). In contrast, larger problem dimension N increases the bound, reflecting the inherent difficulty of high-dimensional estimation. These dependencies are analogous to those observed for graphical Lasso [15]. The additional term $\|\mathbf{\Theta}_0 - \tilde{\mathbf{\Theta}}^{(i)}\|_*^{1/2}$ captures the distance between the true precision and the task-aware matrix $\tilde{\mathbf{\Theta}}^{(i)}$, which can be negligible if the task labels reveal meaningful dependency patterns, but may bias the estimate away from the true precision if such patterns are not aligned with the underlying statistical dependencies. This result justifies the role of the decopling strategy that aims at learning jointly the precision matrix and the PNN weights while remaining principled to the ground truth precision.

4. NUMERICAL RESULTS

We evaluate PNNs over synthetic and real-world datasets with two main objectives: **(O1)** to validate the superior downstream performance of joint optimization compared to two-stage approaches, and **(O2)** to assess the ability of our method to produce informative and sparse precision estimates.

4.1. Baselines and Hyperparameters

We compare four variants of PNN: PNN on the sample precision (**Sample**); PNN on the precision estimated via graphical Lasso

Authorized licensed use limited to: TU Delft Library. Downloaded on May 12,2026 at 10:23:32 UTC from IEEE Xplore. Restrictions apply.

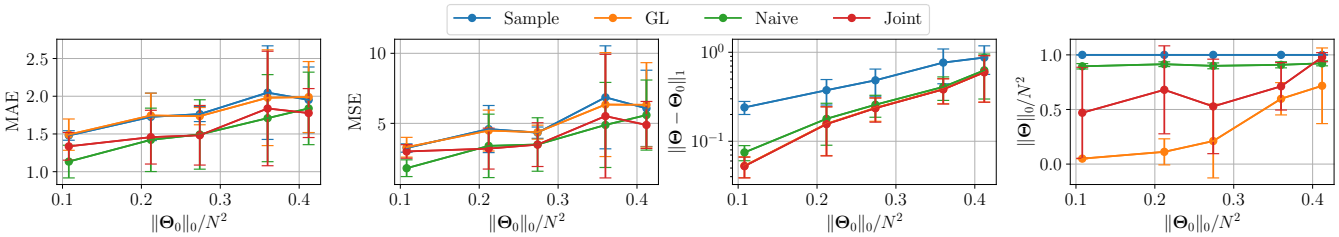


Fig. 2: Regression error (MAE and MSE), precision matrix reconstruction quality in terms of L1 norm and sparsity of estimated precision $\mathbf{\Theta}$ on the synthetic datasets for varying sparsity of the true precision $\mathbf{\Theta}_0$.

(**GL**); PNN with joint precision estimation using the naive formulation in (5) followed by a final soft-thresholding step (**Naive**); and PNN trained with the alternating optimization in Algorithm 1 (**Joint**). On real data, we compare with two additional baselines: **VNN** [7] and PCA projection followed by an MLP regressor (**PCA**). For all models and datasets, hyperparameters are tuned on a validation set over the following ranges: number of layers $L \in \{1, 2, 3\}$, filterbank size $F \in \{8, 16\}$, filter order $K \in \{1, 2, 3\}$, $\lambda_0 \in \{1, 10, 20\}$ where $\lambda = \lambda_0 \sqrt{\log N/T}$, $\alpha = 0.5$, learning rate $\eta = 0.01$, $\gamma = 10$, $I_{\text{epochs}} = 10$, and $I_\Theta = I_{\tilde{\Theta}} = I_h = 20$. Batch normalization is applied to all PNNs. For synthetic data, hyperparameters are selected on the setting with sparsity 0.2 and reused for the other settings. All datasets are split into train/validation/test sets of size 60%/20%/20%, and each experiment is repeated 5 times, reporting mean and standard deviation. All code and additional implementation details are available on GitHub[1].

4.2. Synthetic Data

We generate synthetic datasets by defining a precision matrix $\mathbf{\Theta}_0$ with prescribed sparsity ($\|\mathbf{\Theta}_0\|_0/N^2$) and sampling observations as $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Theta}_0^{-1})$. We stack the samples into a matrix $\mathbf{X} \in \mathbb{R}^{N \times T}$ and create regression targets as $\mathbf{y} = \mathbf{w}^\mathsf{T}\mathbf{X} + \mathbf{z}$, with $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and noise entries $z_i \sim \mathcal{N}(0, \sigma)$, where σ is set to achieve an SNR of 10. We set $N = 20$ and $T = 100$. From Figure 2, **Joint** and **Naive** achieve comparable or better performance than **GL** and **Sample** across all sparsity levels, in terms of both MAE and MSE. **Joint** consistently provides the best results, confirming the benefit of the joint optimization for downstream performance **(O1)**. Regarding precision reconstruction quality, **GL** and **Joint** perform similarly in terms of ℓ_1 error, while **Naive** yields poorer estimates and **Sample** is particularly inaccurate due to the numerous spurious entries **(O2)**. Finally, both **GL** and **Joint** recover sparsity levels close to the ground truth, enabling efficient implementations. In contrast, **Sample** contains no zero entries due to noise, and **Naive** produces very few zeros as a result of its unstable training dynamics.

4.3. Real Data

We further evaluate PNNs on two brain datasets, ADNI and ABIDE, containing cortical thickness measurements derived from MRI scans. ADNI [28] includes patients with Alzheimer's disease and healthy controls. We use the ADNI2 collection, extracting cortical thickness measures from the FreeSurfer outputs [29] available at [2], resulting in $N = 68$ brain regions and $T = 1142$ patients. ABIDE [30] includes patients with autism and healthy controls. We obtain cortical thickness measures following the steps described at [3], retaining the features of the Mindboggle protocol [31], which yields $N = 62$ and $T = 1035$. Our goal is to predict patient age from brain measurements, a problem of interest in neuroscience since discrepancies between brain age and chronological age are indicative of neurodegenerative disease [8].

Method	ADNI		ABIDE	
	MAE	#Zeros	MAE	#Zeros
PCA	10.3±0.7	–	4.41±0.34	–
VNN	6.06±0.20	–	5.81±0.11	–
Sample	5.78±0.10	0±0	5.94±0.02	0±0
GL	5.81±0.06	4546±0	5.80±0.11	2232±0
Joint	**5.31±0.18**	1225±1126	**3.64±0.65**	2771±976

Table 1: Regression error (MAE) and number of zero elements in the estimated precision matrix for the age prediction task from cortical thickness measurements. We do not report sparsity metrics for PCA and VNN as they do not use the precision matrix.

Table 1 reports the results. Overall, **Joint** achieves the best performance in terms of MAE on the age prediction task **(O1)**, while also producing sparse precision estimates **(O2)**, confirming the benefit of the joint estimation for real-world downstream tasks. Both **GL** and **Joint** outperform **Sample** by relying on cleaner precision estimates, further corroborating that sample estimates in real data contain noisy dependencies that hinder predictive performance. **VNN** performs closely to **GL** on ABIDE, but worse on ADNI, as it also operates on the sample covariance which contains spurious correlations. **PCA**, finally, performs well on ABIDE but particularly badly on ADNI, indicating that more complex covariance processing is helpful to achieve consistent performance across settings.

5. CONCLUSION

We introduced PNNs, a class of graph convolutional NNs that leverage sparse estimates of the precision matrix, and we proposed a novel paradigm based on alternating optimization to jointly estimate the NN weights and the sparse precision matrix. We established theoretical guarantees on the stability of the estimated sparse precision with respect to the true precision, considering both the number of available observations and the downstream task. Our numerical experiments on synthetic and real datasets, focusing on age prediction from cortical thickness measurements, demonstrated the effectiveness of the proposed joint learning strategy. Future work will focus on improving the interpretability of the task-aware precision matrices, exploring the stability of PNNs in low-data regimes, and further analyzing the convergence properties of the joint optimization.

[1] https://github.com/andrea-cavallo-98/PNN
[2] https://ida.loni.usc.edu/
[3] http://preprocessed-connectomes-project.org/abide/

Authorized licensed use limited to: TU Delft Library. Downloaded on May 12,2026 at 10:23:32 UTC from IEEE Xplore. Restrictions apply.

6. REFERENCES

[1] X. Li, Y. Zhou, N. Dvornek, M. Zhang, S. Gao, J. Zhuang, D. Scheinost, L. Staib, P. Ventola, and J. Duncan, "Braingnn: Interpretable brain graph neural network for fmri analysis," *Medical Image Analysis*, vol. 74, pp. 102233, 09 2021.

[2] A. Bessadok, M. A. Mahjoub, and I. Rekik, "Graph neural networks in network neuroscience," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5833–5848, 2022.

[3] T. Liao, J. Zhao, Y. Liu, K. Ivanov, J. Xiong, and Y. Yan, "Deep transfer learning with graph neural network for sensor-based human activity recognition," in *2022 IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM)*, 2022, pp. 2445–2452.

[4] D. P Palomar, "Portfolio optimization," *MSc in Financial Mathematics, HKUST,[Online]. Available: https://www. danielppalomar. com/mafs6010r—portfolio-optimization-with-r. html*, 2024.

[5] J. V. Cardoso, J. Ying, and D. P. Palomar, "Algorithms for learning graphs in financial markets," *arXiv preprint arXiv:2012.15410*, 2020.

[6] Ian Jolliffe, *Principal component analysis*, Springer Verlag, New York, 2002.

[7] S. Sihag, G. Mateos, C. McMillan, and A. Ribeiro, "Covariance neural networks," *Conf. Neural Inform. Process. Syst.*, vol. 35, pp. 17003–17016, 2022.

[8] S. Sihag, G. Mateos, C. McMillan, and A. Ribeiro, "Explainable brain age prediction using covariance neural networks," *Conf. Neural Inform. Process. Syst.*, vol. 36, 2024.

[9] S. Sihag, G. Mateos, C. McMillan, and A. Ribeiro, "Transferability of covariance neural networks," *IEEE J. Select. Topics Signal Process.*, vol. 18, no. 2, pp. 199–215, 2024.

[10] A. Cavallo, M. Sabbaqi, and E. Isufi, "Spatiotemporal covariance neural networks," in *Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*. Springer, 2024, pp. 18–34.

[11] A. Cavallo, Z. Gao, and E. Isufi, "Sparse covariance neural networks," *arXiv preprint arXiv:2410.01669*, 2024.

[12] A. Cavallo, M. Navarro, S. Segarra, and E. Isufi, "Fair covariance neural networks," in *IEEE Int. Conf. Acoust., Speech and Signal Process.* IEEE, 2025, pp. 1–5.

[13] Jerome Friedman, "Multivariate adaptive regression splines," *The Ann. of Stat.*, vol. 19, no. 1, pp. 1–67, 1991.

[14] S. L. Lauritzen, *Graphical Models*, vol. 17, Clarendon Press, 1996.

[15] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 12 2007.

[16] S. Rey, T. M. Roddenberry, S. Segarra, and A. G. Marques, "Enhanced graph-learning schemes driven by similar distributions of motifs," *IEEE Trans. Signal Process.*, vol. 71, pp. 3014–3027, 2023.

[17] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Intl. Conf. Knowl. Discovery Data Mining (ACM SIGKDD)*, 2020, pp. 66–74.

[18] S. Rey, V. M. Tenorio, and A. G. Marqués, "Robust graph filter identification and graph denoising from signal observations," *IEEE Trans. Signal Process.*, vol. 71, pp. 3651–3666, 2023.

[19] H. Liu, C.-Y. Yau, and H.-T. Wai, "A two-timescale primal-dual algorithm for decentralized optimization with compression," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2025, pp. 1–5.

[20] H. E Egilmez, E. Pavez, and A. Ortega, "Graph learning from filtered signals: Graph system and diffusion kernel identification," *IEEE Trans. Signal, Inform. Process. Networks*, vol. 5, no. 2, pp. 360–374, 2018.

[21] A. Natali, M. Coutino, and G. Leus, "Topology-aware joint graph filter and edge weight identification for network processes," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2020, pp. 1–6.

[22] 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, Nov. 2020.

[23] V. M. Tenorio, S. Rey, and A. G. Marques, "Robust graph neural network based on graph denoising," in *Asilomar Conf. Signals, Syst., and Computers*, 2023, pp. 578–582.

[24] A. Beck, *First-order methods in optimization*, SIAM, 2017.

[25] M. Navarro, C. Little, G. I. Allen, and Santiago S., "Data augmentation via subgroup mixup for improving fairness," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2024, pp. 7350–7354.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] A. Cavallo, S. Rey, A. G. Marques, and E. Isufi, "Precision neural networks: Joint graph and relational learning," *arXiv preprint arXiv:2509.14821*, 2025.

[28] C. R. Jack Jr, M. A. Bernstein, N. C. Fox, P. Thompson, G. Alexander, D. Harvey, B. Borowski, P. J. Britson, J. Whitwell, C. Ward, et al., "The alzheimer's disease neuroimaging initiative (adni): Mri methods," *J. Magn. Reson. Imaging*, vol. 27, no. 4, pp. 685–691, 2008.

[29] B. Fischl, "Freesurfer," *Neuroimage*, vol. 62, no. 2, pp. 774–781, 2012.

[30] C. Craddock, Y. Benhajali, C. Chu, F. Chouinard, A. Evans, A. Jakab, B. S. Khundrakpam, J. D. Lewis, Q. Li, M. Milham, et al., "The neuro bureau preprocessing initiative: open sharing of preprocessed neuroimaging data and derivatives," *Frontiers in Neuroinformatics*, vol. 7, no. 27, pp. 5, 2013.

[31] A. Klein, S. S. Ghosh, F. S. Bao, J. Giard, Y. Häme, E. Stavsky, N. Lee, B. Rossa, M. Reuter, E. Chaibub Neto, et al., "Mindboggling morphometry of human brains," *PLoS computational biology*, vol. 13, no. 2, pp. e1005350, 2017.

Authorized licensed use limited to: TU Delft Library. Downloaded on May 12,2026 at 10:23:32 UTC from IEEE Xplore. Restrictions apply.