Graphs in Financial Markets

Daniel P. Palomar

The Hong Kong University of Science and Technology (HKUST)

Keynote Talk

Graph Signal Processing Workshop (GSPW 2026)
Madrid, Spain

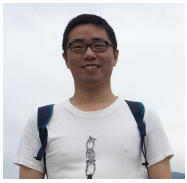
June 8–10, 2026

Outline

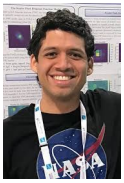
Graph Research Collaborators

Licheng Zhao

Sandeep Kumar



Jiaxi Ying



Vinícius Cardoso

Xiwen Wang

Amirhossein Javaheri

Arash Amini

Farokh Marvasti

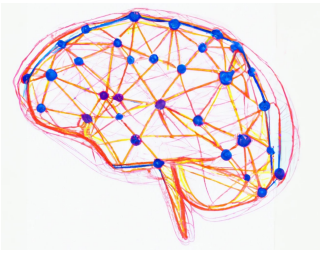
Andrei Buciulea

Antonio G. Marques

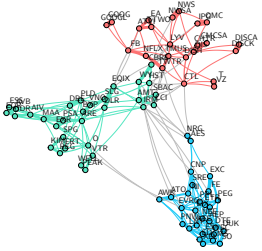
Outline

Graphs Everywhere

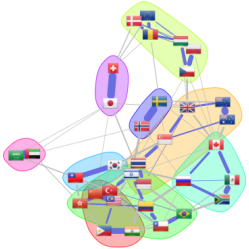
Social network



Brain connectivity



Financial stocks

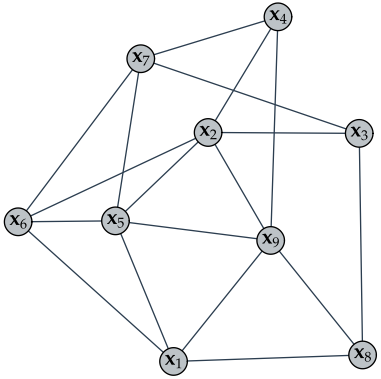


Foreign exchange

Graphical Models

Graphical models are a way to represent and visualize the **relationships** between entities:

- **nodes** correspond to the entities (variables)
- **edges** encode the relationships between entities (dependencies between the variables)
- **edge weights** $\boldsymbol{W}$ encode the strength of the relationships



Graph Matrices

The **adjacency matrix** $\boldsymbol{W}$ directly characterizes a graph: W_{ij} contains the weight of the edge between nodes i and j.

The **degree matrix** $\boldsymbol{D}$ is a diagonal matrix containing the node degrees (row sums of $\boldsymbol{W}$):

$$\boldsymbol{D} = \mathrm{Diag}(\boldsymbol{W}\mathbf{1}).$$

The **Laplacian** $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the key matrix in graph signal processing:

- *Physical interpretation*: measures the **smoothness** of a graph signal $\boldsymbol{x} = (x_1, \ldots, x_p)$:

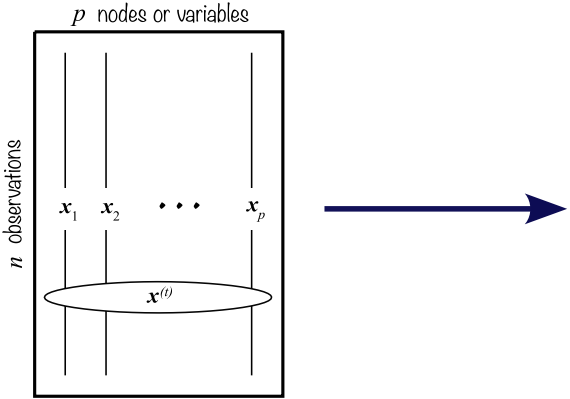
$$\boldsymbol{x}^\top \boldsymbol{L}\, \boldsymbol{x} = \frac{1}{2} \sum_{i,j} W_{ij}(x_i - x_j)^2$$

- *Mathematical properties*: symmetric and positive semidefinite ($\boldsymbol{L} \succeq \mathbf{0}$); singular with $\boldsymbol{L}\mathbf{1} = \mathbf{0}$

Outline

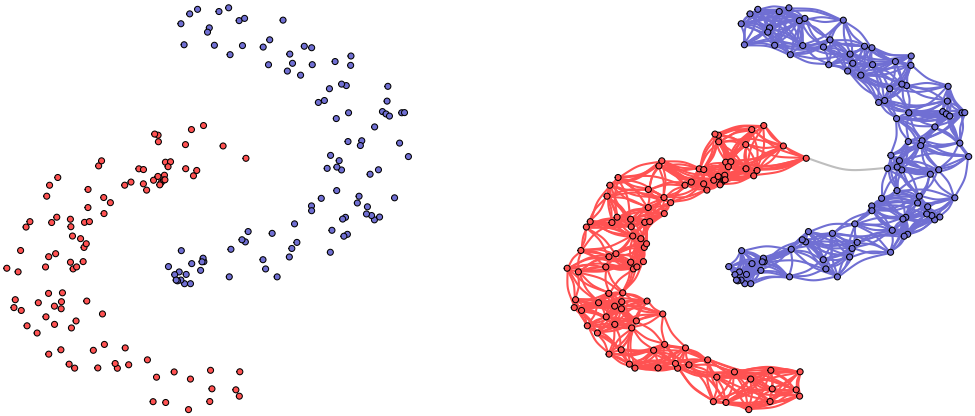
Learning Graphs from Data

- Given a data matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p] \in \mathbb{R}^{T \times p}$, each column $\boldsymbol{x}_i \in \mathbb{R}^T$ contains T observations from node i (and each row is one observation of the **graph signal**).
- The goal is to obtain a **graph representation** of the data.



Graph Learning Illustration

Example of learning a graph from a dataset with points in $\mathbb{R}^2$:



Overview references: (Mateos et al. 2019) and (Dong et al. 2019).

Learning Graphs from Smooth Signals

- Since $\boldsymbol{x}^\mathsf{T}\boldsymbol{L}\boldsymbol{x}$ measures the smoothness of a graph signal $\boldsymbol{x}$, the total smoothness over T observations collected along the rows of matrix $\boldsymbol{X}$ is $\mathrm{Tr}(\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^\mathsf{T})$.
- Learn the graph by finding the Laplacian that makes the signals smooth (Kalofolias 2016):

$$\begin{aligned} \underset{\boldsymbol{L}\succeq\boldsymbol{0}}{\text{minimize}} \quad & \mathrm{Tr}\Big(\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^\mathsf{T}\Big) - \alpha\,\boldsymbol{1}^\mathsf{T}\log(\mathrm{diag}(\boldsymbol{L})) + \tfrac{\beta}{2}\|\boldsymbol{L}\|_{F,\text{off}}^2 \\ \text{subject to} \quad & \boldsymbol{L}\boldsymbol{1} = \boldsymbol{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall\, i \neq j. \end{aligned}$$

  where $\mathrm{diag}(\boldsymbol{L})$ contains the node degrees, so $-\log(\mathrm{diag}(\boldsymbol{L}))$ penalizes zero degrees (no isolated nodes); and $\|\boldsymbol{L}\|_{F,\text{off}}^2$ penalizes nodes with very few connections.
- The same formulation can be equivalently rewritten in terms of the adjacency matrix $\boldsymbol{W}$ instead of $\boldsymbol{L}$.
- Other formulations: (Dong et al. 2015) and (Nie et al. 2016).

Graphical Lasso

- **MLE of the covariance matrix Σ** from T observations (with sample covariance $\boldsymbol{S}$):

$$\underset{\boldsymbol{\Sigma} \succ \mathbf{0}}{\text{maximize}} \quad \log \det(\boldsymbol{\Sigma}^{-1}) - \text{Tr}(\boldsymbol{\Sigma}^{-1}\boldsymbol{S}).$$

- The **precision matrix** $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$ encodes partial correlations: $\Theta_{ij} = 0$ implies nodes i and j are conditionally independent given all others — this defines the graph edge structure.

- **Graphical LASSO (GLASSO)** adds an ℓ_1 penalty to promote sparsity in $\boldsymbol{\Theta}$ (Meinshausen and Bühlmann 2006):

$$\underset{\boldsymbol{\Theta} \succ \mathbf{0}}{\text{maximize}} \quad \log \det(\boldsymbol{\Theta}) - \text{Tr}(\boldsymbol{\Theta}\boldsymbol{S}) - \rho\|\boldsymbol{\Theta}\|_{1,\text{off}},$$

where ρ controls the sparsity level.

Learning Graphs from a Gaussian Markov Random Field

- Modeling graph signals as a **Gaussian Markov Random Field (GMRF)**, the MLE of the graph Laplacian is (Lake and Tenenbaum 2010; Egilmez, Pavez, and Ortega 2017; Zhao et al. 2019):

$$\begin{array}{ll} \underset{\boldsymbol{L}\succeq\mathbf{0}}{\text{maximize}} & \log \text{gdet}(\boldsymbol{L}) - \text{Tr}(\boldsymbol{S}\boldsymbol{L}) - \rho\|\boldsymbol{L}\|_{1,\text{off}} \\ \text{subject to} & \boldsymbol{L}\mathbf{1} = \mathbf{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall\, i \neq j, \end{array}$$

  where $\boldsymbol{S}$ is the sample covariance matrix.
- Surprisingly, the ℓ_1-norm regularization $\|\boldsymbol{L}\|_{1,\text{off}}$ does **not** promote sparsity — a counter-intuitive result (Ying, M. Cardoso, and Palomar 2020, 2021).
- Instead, one must use a **concave regularizer** to achieve sparse graphs.
- A concave regularizer leads to a nonconvex problem, but it can be solved iteratively via **majorization-minimization (MM)**, yielding a **reweighted ℓ_1-norm** method (Kumar et al. 2020; Ying, M. Cardoso, and Palomar 2020).

Trick: Laplacian Operator*

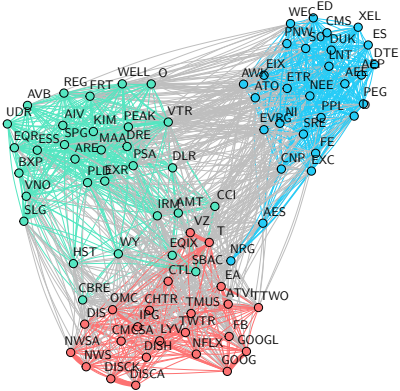
- $\boldsymbol{L}$ contains p^2 elements but the structural constraints reduce the effective degrees of freedom to $p(p-1)/2$. A more compact representation uses the vector $\boldsymbol{w} \in \mathbb{R}_+^{p(p-1)/2}$ of edge weights (off-diagonal upper-triangular entries of $\boldsymbol{L}$).
- The linear **Laplacian operator** $\mathcal{L}(\boldsymbol{w})$ reconstructs the full Laplacian — e.g., for a 3-node graph:

$$\mathcal{L}(\boldsymbol{w}) = \begin{bmatrix} w_1 + w_2 & -w_1 & -w_2 \\ -w_1 & w_1 + w_3 & -w_3 \\ -w_2 & -w_3 & w_2 + w_3 \end{bmatrix}.$$

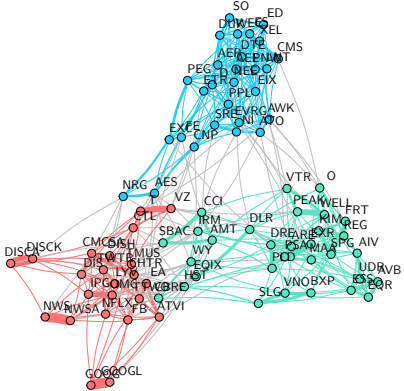
- $\boldsymbol{L} = \mathcal{L}(\boldsymbol{w})$ automatically satisfies all structural constraints $\boldsymbol{L}\mathbf{1} = \mathbf{0}$, $L_{ij} = L_{ji} \leq 0$, $\forall\, i \neq j$, so the optimization can be carried out directly over $\boldsymbol{w} \geq \mathbf{0}$ (Kumar et al. 2019, 2020).

Numerical Experiments: Effect of Sparsity Regularizer

S&P 500 stocks (3 sectors: Industrials, Consumer Staples, Energy; 2016–2019):



GMRF graph (ℓ_1-norm)



GMRF graph (reweighted ℓ_1-norm)

▶ *The ℓ_1-norm produces a dense graph — motivating the need for a concave (reweighted ℓ_1) regularizer.*

Outline

Graph Shift Operator and Graph Stationarity

- The **Graph Shift Operator (GSO)** $\boldsymbol{S} \in \mathbb{R}^{p \times p}$ encodes the graph topology: $S_{ij} \neq 0$ iff $i = j$ or nodes i and j are connected by an edge. Common choices: adjacency matrix $\boldsymbol{W}$ or Laplacian $\boldsymbol{L}$.
- A **graph filter** is a polynomial of the GSO: $\boldsymbol{H} = \sum_{k=0}^{K} h_k \boldsymbol{S}^k$. It operates locally — each node aggregates information from its K-hop neighborhood.
- **Physical interpretation:** since $\boldsymbol{S}^k$ propagates a signal k hops through the graph, $\boldsymbol{H}$ captures **multi-hop (delayed) interactions** among nodes — as opposed to an unstructured precision matrix $\boldsymbol{\Theta}$, which only models instantaneous pairwise interactions.
- A graph signal $\boldsymbol{x}$ is **stationary on** $\boldsymbol{S}$ if its covariance $\boldsymbol{\Sigma}$ commutes with $\boldsymbol{S}$: $\boldsymbol{\Sigma}\boldsymbol{S} = \boldsymbol{S}\boldsymbol{\Sigma}$. Equivalently, $\boldsymbol{\Sigma}$ and $\boldsymbol{S}$ share the same eigenvectors, i.e., $\boldsymbol{\Sigma}$ is a polynomial of $\boldsymbol{S}$:

$$\boldsymbol{\Sigma} = \boldsymbol{H}^2 = \left(\sum_{k=0}^{K} h_k \boldsymbol{S}^k \right)^2 = p(\boldsymbol{S}).$$

The PGL Formulation

- **Recall:** the Graphical LASSO estimates the sparse precision matrix $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$:

$$\underset{\boldsymbol{\Theta} \succ \mathbf{0}}{\text{maximize}} \quad \log\det(\boldsymbol{\Theta}) - \text{Tr}(\boldsymbol{\Theta}\hat{\boldsymbol{\Sigma}}^{\text{SCM}}) - \rho\|\boldsymbol{\Theta}\|_{1,\text{off}},$$

  where $\hat{\boldsymbol{\Sigma}}^{\text{SCM}}$ is the sample covariance matrix; but ignores any underlying graph structure.

- **PGL** jointly estimates $\boldsymbol{\Theta}$ and the GSO $\boldsymbol{S}$ under stationarity — the commutativity constraint $\boldsymbol{\Theta}\boldsymbol{S} = \boldsymbol{S}\boldsymbol{\Theta}$ encodes that $\boldsymbol{\Theta}$ is a polynomial of $\boldsymbol{S}$ (Buciulea et al. 2025):

$$\begin{aligned} \underset{\boldsymbol{\Theta} \succ \mathbf{0},\, \boldsymbol{S} \in \mathcal{S}}{\text{maximize}} \quad & \log\det(\boldsymbol{\Theta}) - \text{Tr}(\hat{\boldsymbol{\Sigma}}^{\text{SCM}}\boldsymbol{\Theta}) - \rho\|\boldsymbol{S}\|_0 \\ \text{subject to} \quad & \boldsymbol{\Theta}\boldsymbol{S} = \boldsymbol{S}\boldsymbol{\Theta}, \end{aligned}$$

  where $\mathcal{S}$ encodes structural constraints on $\boldsymbol{S}$ (symmetric, non-negative off-diagonal, zero diagonal, $\boldsymbol{S}\mathbf{1} \geq \mathbf{1}$).

PGL Algorithm

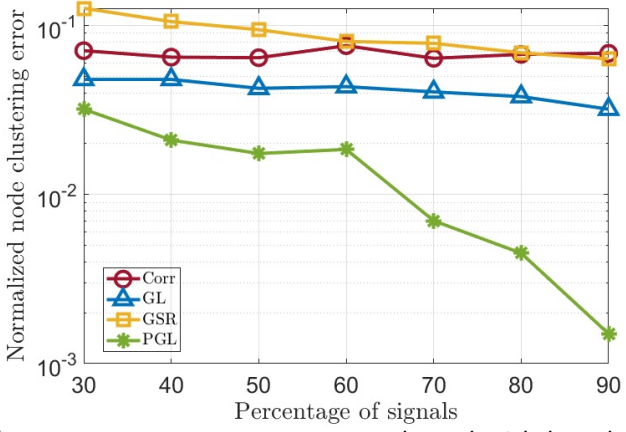
- **Two challenges:** the ℓ_0 sparsity penalty and the commutativity constraint $\boldsymbol{\Theta S} = \boldsymbol{S\Theta}$ are both non-convex.

- **Solution strategy:** relax ℓ_0 to a reweighted ℓ_1, then apply **Majorization-Minimization (MM)** (Sun, Babu, and Palomar 2017) — alternating between two closed-form updates (Buciulea et al. 2025):
  - Update $\boldsymbol{\Theta}$ with $\boldsymbol{S}$ fixed: $\boldsymbol{\Theta}^\star = \boldsymbol{V}\,\mathrm{diag}\Big(\frac{1}{2}\Big(-\hat{\boldsymbol{\lambda}} + \sqrt{\hat{\boldsymbol{\lambda}}^2 + 4}\Big)\Big)\,\boldsymbol{V}^{\mathsf{T}}$, where $\hat{\boldsymbol{\lambda}}$ are the eigenvalues of $\hat{\boldsymbol{\Sigma}}^{\mathrm{SCM}}$ in the basis of $\boldsymbol{S}$.
  - Update $\boldsymbol{S}$ with $\boldsymbol{\Theta}$ fixed: element-wise soft-thresholding in the eigenbasis of $\boldsymbol{\Theta}$.

- **Guarantees:** converges to a stationary point; a fast variant (GST-fast) reduces computational complexity significantly.

PGL Numerical Experiments: Stock Clustering

S&P 500 data (40 stocks, 4 sectors): clustering error vs. fraction of data used for graph learning (Buciulea et al. 2025):



▶ *PGL recovers the sector structure more accurately and with less data than GL, GSR, and correlation baselines.*

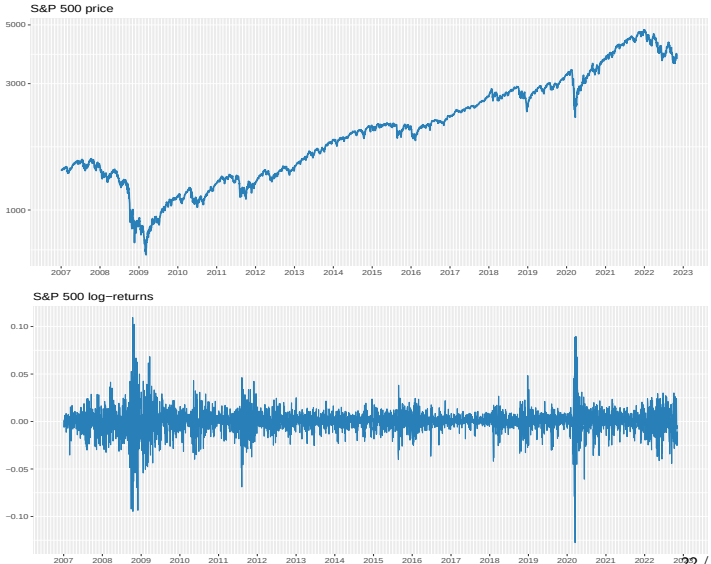
Outline

Prices and Returns

- **Price:** p_t (discrete time periods: minutes to years)
- **Log-price:** $y_t \triangleq \log p_t$
- **Linear return:** $r_t^{\text{lin}} \triangleq \dfrac{p_t - p_{t-1}}{p_{t-1}}$
- **Log-return:** $r_t \triangleq y_t - y_{t-1} = \log\left(\dfrac{p_t}{p_{t-1}}\right)$
- **Random walk model:**

$$y_t = \mu + y_{t-1} + \epsilon_t \Rightarrow r_t = \mu + \epsilon_t$$

(Ruppert 2010; Palomar 2025, ch.2)



Stylized Facts

Empirical regularities observed across instruments, markets, and time periods (Cont 2001):

Absence of Return Autocorrelations

- Returns are largely unpredictable (efficient-market hypothesis, Fama 1970)
- But $|r_t|$ and r_t^2 *are* autocorrelated — nonlinear dependence persists

Volatility Clustering / Lack of Stationarity

- Large moves tend to follow large moves; variance changes over time (Mandelbrot 1963)

$\Rightarrow$ *Addressed in Section 7: Learning Time-Varying Graphs*

Heavy Tails

- Distributions exhibit much heavier tails than Gaussian; more frequent extreme events

$\Rightarrow$ *Addressed in Section 5: Learning Heavy-Tailed Graphs*

Correlation Structure

- Assets co-move: pairwise correlations are predominantly positive and reveal market factor structure

$\Rightarrow$ *Addressed in Section 6: Learning Structured Graphs*

Stylized Fact 1: Heavy Tails

Gaussian distribution (fully characterized by mean and variance):

$$f(x) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

Skewness (asymmetry) and **kurtosis** (tail thickness):

$$\gamma_1 = \frac{\mathbb{E}[(r-\mu)^3]}{\sigma^3}, \qquad \kappa = \frac{\mathbb{E}[(r-\mu)^4]}{\sigma^4}$$

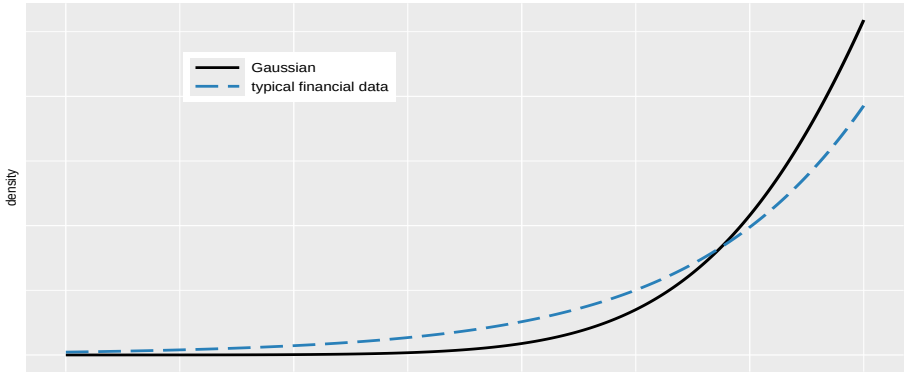
- Gaussian: $\gamma_1 = 0$, $\kappa = 3$ (excess kurtosis $= 0$)

Heavy tails: kurtosis > 3 — tail decay slower than Gaussian

- More frequent extreme events than Gaussian predicts
- Financial returns exhibit heavier tails, especially on the left (large losses)

(Ruppert 2010; Palomar 2025, ch.2)

Stylized Fact 1: Heavy Tails (cont.)

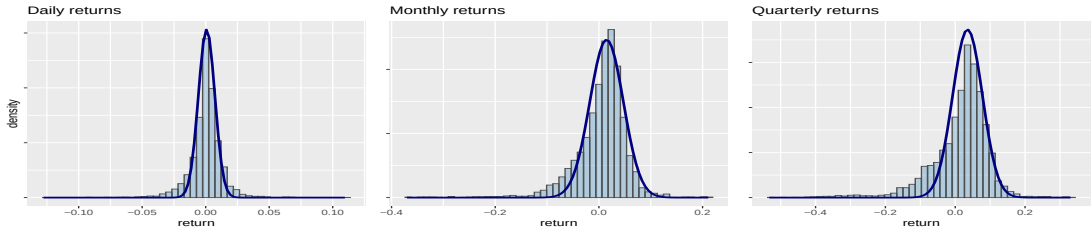


Implications for finance:

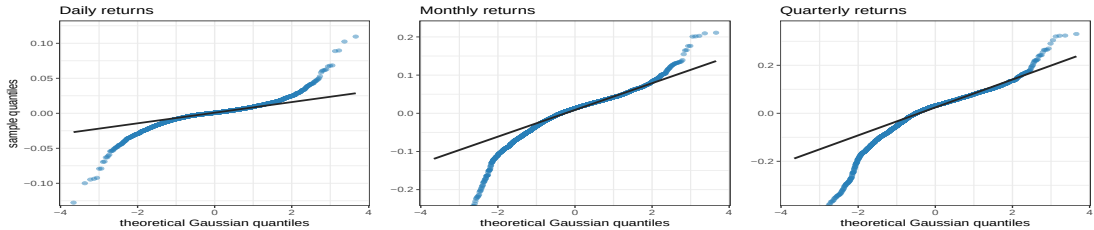
- **Risk underestimation**: Gaussian models underestimate the probability of crashes and large losses
- **Portfolio optimization**: ignoring heavy tails leads to portfolios that are too aggressive

Stylized Fact 1: Heavy Tails (S&P 500)

S&P 500 log-return histograms at increasing frequencies:



Q-Q plots vs. Gaussian (deviations from the line = heavy tails):



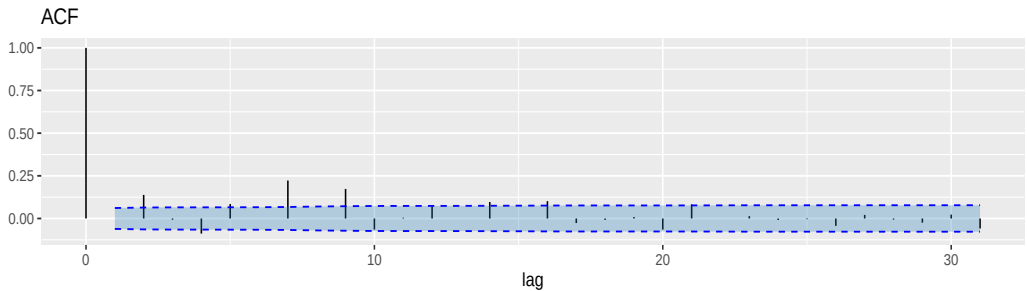
Stylized Fact 2: Volatility Clustering

Efficient-Market Hypothesis (EMH)

- Share prices reflect **all available information** at all times (Fama, 1970)
- Neither technical nor fundamental analysis can consistently generate excess returns
- In practice: financial returns behave approximately as **i.i.d.**.

Linear temporal structure: ACF of returns

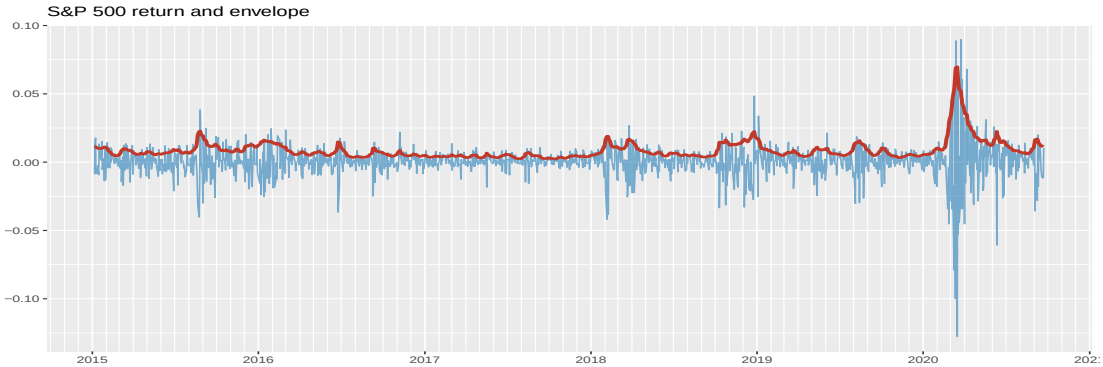
- ACF measures linear dependence between r_t and r_{t-k}
- EMH predicts no significant autocorrelation in returns
- Empirically confirmed for S&P 500: ACF lags lie within noise bands



Stylized Fact 2: Volatility Clustering (cont.)

Nonlinear structure: the volatility envelope clusters

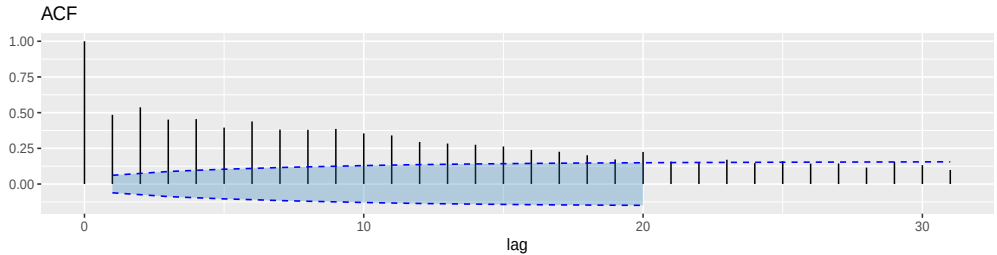
- Large changes tend to follow large changes (high volatility periods)
- Small changes tend to follow small changes (calm periods)
- Documented by Mandelbrot (1963) and Fama (1965)
- Modeled by ARCH/GARCH family of processes



Stylized Fact 2: Volatility Clustering (cont.)

ACF of $|r_t|$: nonlinear dependence persists

- Unlike ACF of r_t, the ACF of $|r_t|$ is significant at many lags
- Confirms that volatility — not returns — carries temporal structure



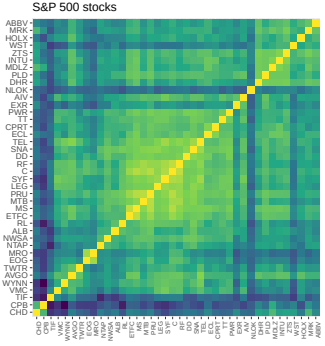
Implications for finance:

- **Volatility is forecastable**: ARCH/GARCH models exploit this for risk management
- **Non-stationarity**: the statistics and distribution of financial returns change over time — addressed later when learning time-varying graphs

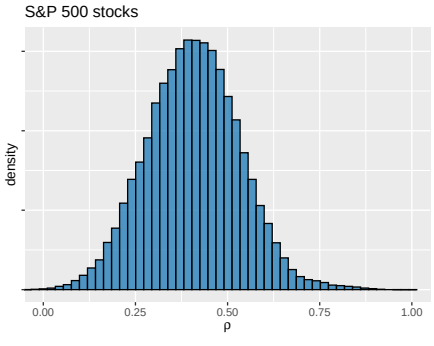
Stylized Fact 3: Correlation Structure

Assets are not independent — pairwise correlations are predominantly positive

- Stocks co-move: when the market rises (falls), most stocks rise (fall) together
- Ignoring correlations leads to severe underestimation of portfolio risk



Correlation matrix of S&P 500 stocks



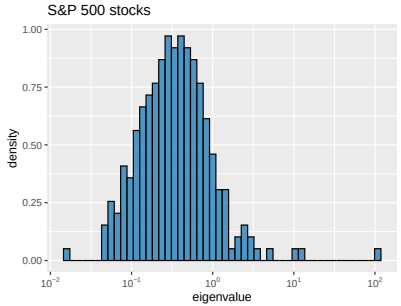
Distribution of pairwise correlations

(Cont 2001; Palomar 2025, ch.2)

Stylized Fact 3: Correlation Structure (cont.)

Factor model structure: one dominant eigenvalue

- Eigenvalues of the correlation matrix reveal the factor structure of the market
- One very large eigenvalue $\gg$ all others: the **market factor** drives most of the variance
- Remaining smaller eigenvalues carry asset-specific (idiosyncratic) structure



Implication: after removing the dominant market factor, the residual correlations reflect genuine pairwise relationships between stocks — naturally modeled as a **graph**

(Palomar 2025, ch.2)

Implications for Graph Learning

The three stylized facts jointly motivate the models developed in this talk:

- **Heavy tails**: the Gaussian distribution is inadequate — a **heavy-tailed distribution model** is needed for robust graph learning (and perhaps even asymmetric)
- **Volatility clustering**: returns are not i.i.d. over time — a **time-varying graph** model is required to track the evolving market structure
- **Correlation structure**: pairwise asset relationships are non-trivial and structured (positive, low-rank, and with assets naturally **clustered** into groups of high correlation) — a **graph** is a natural and interpretable representation (connecting later to structured graph learning)

$\Rightarrow$ We need graph learning algorithms that handle **heavy tails**, **clustered assets**, and **time variation**

Outline

Heavy-Tailed Graph Model

- Recall that the **Gaussian model** $f(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2} \exp\Big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\Big)$ leads to the MLE problem for the graph Laplacian:

$$\begin{array}{ll} \underset{\boldsymbol{L}\succeq\boldsymbol{0}}{\text{maximize}} & \log \operatorname{gdet}(\boldsymbol{L}) - \operatorname{Tr}(\boldsymbol{L}\boldsymbol{S}) \\ \text{subject to} & \boldsymbol{L}\boldsymbol{1} = \boldsymbol{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall\, i \neq j, \end{array}$$

  where $\boldsymbol{S} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{x}^{(t)}(\boldsymbol{x}^{(t)})^{\mathsf{T}}$ is the sample covariance matrix.

- **Student-t model** $f(\boldsymbol{x}) \propto |\boldsymbol{\Sigma}|^{-1/2}\Big(1 + \frac{1}{\nu}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\Big)^{-(p+\nu)/2}$ with degrees of freedom ν (heavier tails as $\nu \downarrow$) leads to:

$$\begin{array}{ll} \underset{\boldsymbol{L}\succeq\boldsymbol{0}}{\text{maximize}} & \log \operatorname{gdet}(\boldsymbol{L}) - \dfrac{p+\nu}{T}\displaystyle\sum_{t=1}^{T} \log\Big(1 + \frac{1}{\nu}(\boldsymbol{x}^{(t)})^{\mathsf{T}}\boldsymbol{L}\,\boldsymbol{x}^{(t)}\Big) \\ \text{subject to} & \boldsymbol{L}\boldsymbol{1} = \boldsymbol{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall\, i \neq j. \end{array}$$

Majorization-Minimization (MM) Algorithm

- The heavy-tailed formulation is **nonconvex**. The **MM framework** (Sun, Babu, and Palomar 2017) upper-bounds each log term:

$$\log\Big(1+\tfrac{1}{\nu}(\boldsymbol{x}^{(t)})^\mathsf{T}\boldsymbol{L}\boldsymbol{x}^{(t)}\Big) \le \text{const} + \frac{(\boldsymbol{x}^{(t)})^\mathsf{T}\boldsymbol{L}\boldsymbol{x}^{(t)}}{\nu+(\boldsymbol{x}^{(t)})^\mathsf{T}\boldsymbol{L}_0\boldsymbol{x}^{(t)}}.$$

- This yields a sequence of **Gaussianized** problems ($k=1,2,\ldots$):

$$\begin{array}{ll}\underset{\boldsymbol{L}\succeq\boldsymbol{0}}{\text{maximize}} & \log\text{gdet}(\boldsymbol{L}) - \text{Tr}(\boldsymbol{L}\boldsymbol{S}^k)\\ \text{subject to} & \boldsymbol{L}\boldsymbol{1}=\boldsymbol{0}, \quad L_{ij}=L_{ji}\le 0, \quad \forall\, i\ne j,\end{array}$$

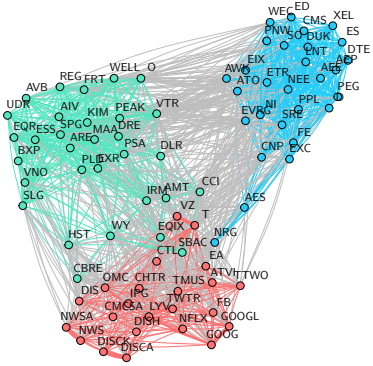
where $\boldsymbol{S}^k = \frac{1}{T}\sum_{t=1}^T w_t^k\,\boldsymbol{x}^{(t)}(\boldsymbol{x}^{(t)})^\mathsf{T}$ is a weighted sample covariance with weights (M. Cardoso, Ying, and Palomar 2021)

$$w_t^k = \frac{p+\nu}{\nu+(\boldsymbol{x}^{(t)})^\mathsf{T}\boldsymbol{L}^k\boldsymbol{x}^{(t)}}.$$

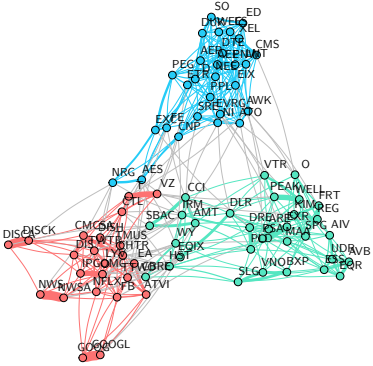
(Palomar 2025, ch. 5)

Numerical Experiments: Gaussian vs. Heavy-Tailed Graphs with Stocks

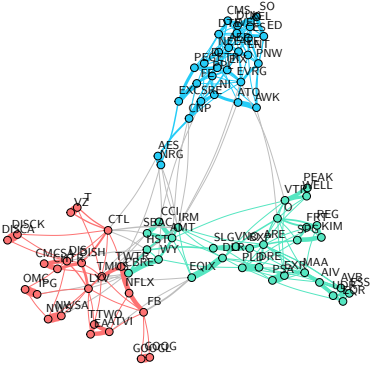
S&P 500 stocks (3 sectors: Industrials, Consumer Staples, Energy; 2016–2019):



GMRF graph (ℓ_1-norm)



GMRF graph (reweighted ℓ_1-norm)

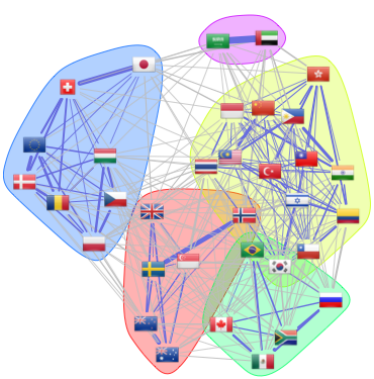


Heavy-tailed MRF graph

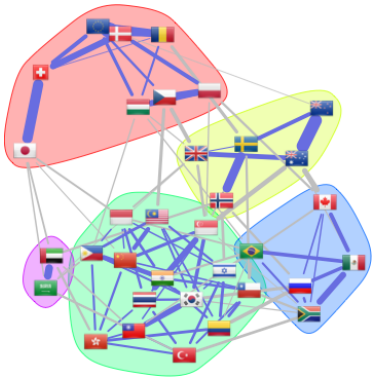
▶ *Heavy-tailed graphs better capture the actual structure of financial data — fewer spurious edges, cleaner sector clusters.*

Numerical Experiments: Gaussian vs. Heavy-Tailed Graphs with FX

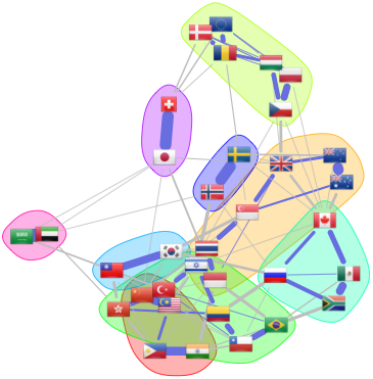
FX currencies (COVID period, 2020):



GMRF graph (ℓ_1-norm)



GMRF graph (reweighted ℓ_1-norm)

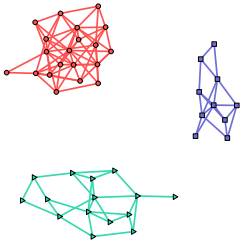


Heavy-tailed MRF graph

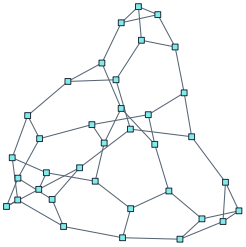
▶ *Heavy-tailed graphs highlight geographically close currency pairs (e.g., HKD/CNY, TWD/KRW, PLN/CZK) — invisible in Gaussian graphs.*

Outline

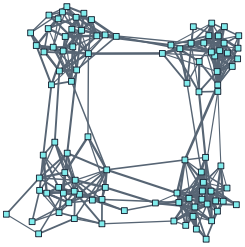
Structured Graphs



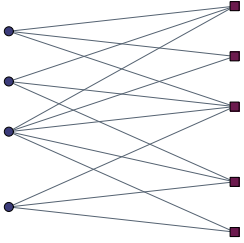
Multi-component



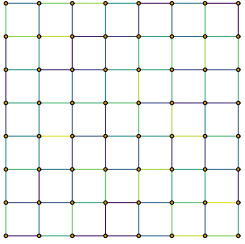
Regular



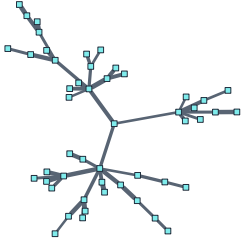
Modular



Bipartite



Grid



Tree

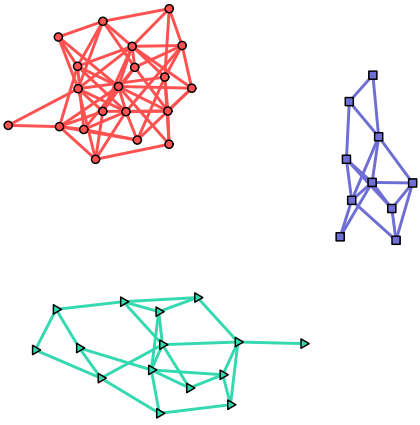
Learning Structured Graphs via Eigenvalues

- Learning graphs with a **specific structure** is generally NP-hard; no universal algorithm exists.
- Simple structures (regular, grid): enforced by fixing elements of $\boldsymbol{L}$ or controlling node degrees.
- Complex structures: characterized by **spectral properties** of $\boldsymbol{L}$ or $\boldsymbol{W}$ — enforced in the learning problem:
  - k-**component graph:** $\boldsymbol{L}$ has k zero eigenvalues (low-rank structure).
  - **Bipartite graph:** $\boldsymbol{W}$ has eigenvalues symmetric around zero.

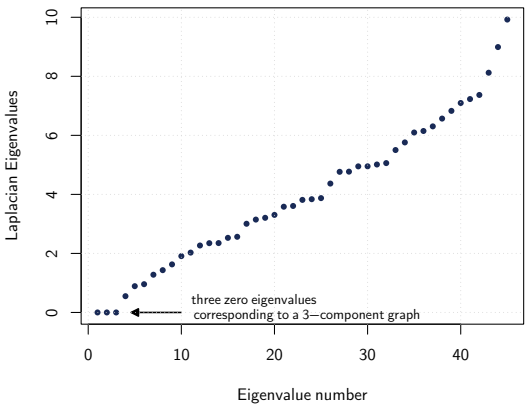
(Palomar 2025, ch. 5)

Example of k-Component Graph

3-component graph (3 clusters) along Laplacian matrix eigenvalues (3 zero eigenvalues):



Graph



Laplacian matrix eigenvalues

Learning k-Component Graphs

- Enforce low-rank structure via **Ky Fan's theorem**:

$$\sum_{i=1}^{k} \lambda_i(\boldsymbol{L}) = \min_{\boldsymbol{F} \in \mathbb{R}^{p \times k}:\, \boldsymbol{F}^\top \boldsymbol{F} = \boldsymbol{I}} \operatorname{Tr}(\boldsymbol{F}^\top \boldsymbol{L} \boldsymbol{F}),$$

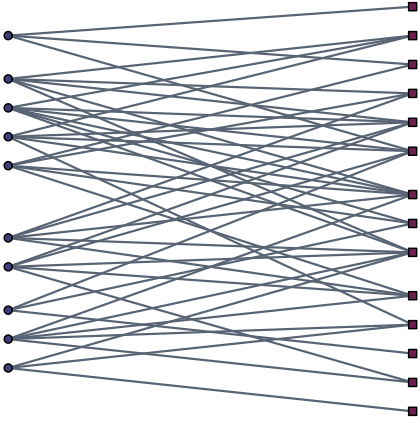
  so it suffices to add the penalty $\operatorname{Tr}(\boldsymbol{F}^\top \boldsymbol{L} \boldsymbol{F})$ to the objective.
- Combining with the heavy-tailed MM framework yields a sequence of Gaussianized problems ($k = 1, 2, \dots$) (Cardoso et al. 2021):

$$\begin{array}{ll} \underset{\boldsymbol{L} \succeq \boldsymbol{0},\, \boldsymbol{F}}{\text{maximize}} & \log \operatorname{gdet}(\boldsymbol{L}) - \operatorname{Tr}(\boldsymbol{L}\boldsymbol{S}^k) - \gamma \operatorname{Tr}(\boldsymbol{F}^\top \boldsymbol{L} \boldsymbol{F}) \\ \text{subject to} & \boldsymbol{L}\boldsymbol{1} = \boldsymbol{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall\, i \neq j, \\ & \boldsymbol{F}^\top \boldsymbol{F} = \boldsymbol{I}, \end{array}$$

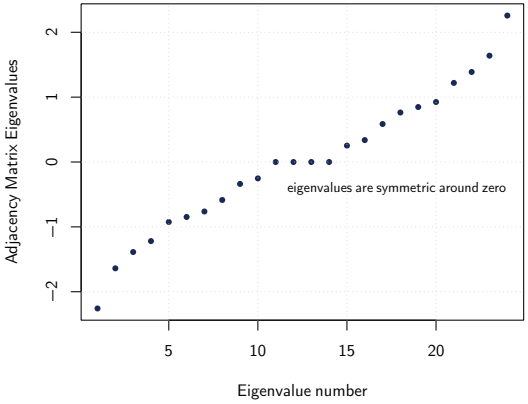
  where $\boldsymbol{S}^k$ is the weighted sample covariance from the MM iteration.
- Solved efficiently by alternating between updates of $\boldsymbol{L}$ (with $\boldsymbol{F}$ fixed) and $\boldsymbol{F}$ (eigenvectors of $\boldsymbol{L}$).

Example of Bipartite Graph

Bipartite graph along adjacency matrix eigenvalues (symmetric around zero):



Graph



Adjacency matrix eigenvalues

Learning Bipartite Graphs

- Two types of nodes with edges only between types — useful to learn connections between individual assets and sector indices.
- Enforce bipartite structure by constraining $\boldsymbol{L}$ to have the block form (with $\boldsymbol{B} \in \mathbb{R}_+^{r \times q}$ containing edge weights):

$$\boldsymbol{L} = \begin{bmatrix} \mathrm{Diag}(\boldsymbol{B}\mathbf{1}) & -\boldsymbol{B} \\ -\boldsymbol{B}^\mathsf{T} & \mathrm{Diag}(\boldsymbol{B}^\mathsf{T}\mathbf{1}) \end{bmatrix}.$$

- The heavy-tailed MM framework yields a sequence of Gaussianized problems ($k = 1, 2, \dots$) (Cardoso et al. 2022):

$$\begin{array}{ll} \underset{\boldsymbol{L} \succeq \mathbf{0},\, \boldsymbol{B}}{\text{maximize}} & \log \mathrm{gdet}(\boldsymbol{L}) - \mathrm{Tr}(\boldsymbol{L}\boldsymbol{S}^k) \\ \text{subject to} & \boldsymbol{L} = \begin{bmatrix} \mathrm{Diag}(\boldsymbol{B}\mathbf{1}) & -\boldsymbol{B} \\ -\boldsymbol{B}^\mathsf{T} & \mathrm{Diag}(\boldsymbol{B}^\mathsf{T}\mathbf{1}) \end{bmatrix}, \\ & \boldsymbol{B} \geq \mathbf{0}. \end{array}$$

Learning k-Component Bipartite Heavy-Tailed Graphs

- Combine the two previous structures: a k-component graph with bipartite connectivity.

- This leads to a sequence of Gaussianized problems ($k = 1, 2, \ldots$) with both $\boldsymbol{F}$ and $\boldsymbol{B}$ as additional variables (M. Cardoso, Ying, and Palomar 2022):

$$\begin{aligned} \underset{\boldsymbol{L} \succeq \boldsymbol{0},\, \boldsymbol{F},\, \boldsymbol{B}}{\text{maximize}} \quad & \log \operatorname{gdet}(\boldsymbol{L}) - \operatorname{Tr}(\boldsymbol{L}\boldsymbol{S}^k) - \gamma \operatorname{Tr}(\boldsymbol{F}^\mathsf{T}\boldsymbol{L}\boldsymbol{F}) \\ \text{subject to} \quad & \boldsymbol{L} = \begin{bmatrix} \operatorname{Diag}(\boldsymbol{B}\boldsymbol{1}) & -\boldsymbol{B} \\ -\boldsymbol{B}^\mathsf{T} & \operatorname{Diag}(\boldsymbol{B}^\mathsf{T}\boldsymbol{1}) \end{bmatrix}, \\ & \boldsymbol{B} \geq \boldsymbol{0}, \quad \boldsymbol{F}^\mathsf{T}\boldsymbol{F} = \boldsymbol{I}. \end{aligned}$$

Learning Graphs with Arbitrary Structures

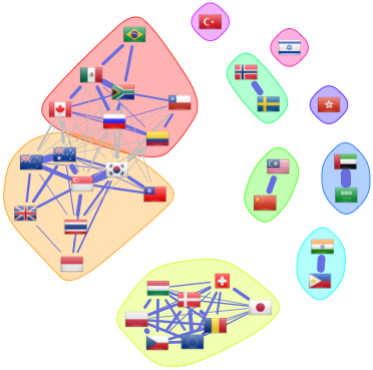
- Structural properties of many graph classes can be characterized by **spectral constraints**:
  - constraints on eigenvalues of the Laplacian matrix $\boldsymbol{L}$
  - constraints on eigenvalues of the adjacency matrix $\boldsymbol{W}$
- **General formulation** with spectral constraint (Kumar et al. 2019, 2020):

$$\begin{array}{ll} \underset{\boldsymbol{L}\succeq\boldsymbol{0}}{\text{maximize}} & \log \operatorname{gdet}(\boldsymbol{L}) - \operatorname{Tr}(\boldsymbol{L}\boldsymbol{S}) - \rho\|\boldsymbol{L}\|_{0,\text{off}} \\ \text{subject to} & \boldsymbol{L}\boldsymbol{1} = \boldsymbol{0}, \quad L_{ij} = L_{ji} \leq 0, \quad \forall\, i \neq j, \\ & \boldsymbol{\lambda}(\boldsymbol{L}) \in \mathcal{S}_{\boldsymbol{\lambda}}, \end{array}$$

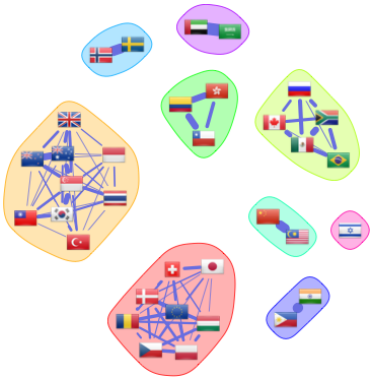
  where $\mathcal{S}_{\boldsymbol{\lambda}}$ encodes the desired spectral structure (e.g., k zero eigenvalues, symmetric eigenvalues).
- Nonconvex problem, but efficient algorithms exist based on MM and the Laplacian operator trick.

Numerical Experiments: Gaussian vs. Heavy-Tailed k-Component Graphs

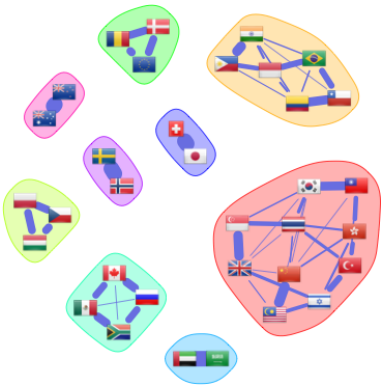
FX currencies (COVID period, 2020):



GMRF graph (ℓ_1-norm)



GMRF graph (reweighted ℓ_1-norm)

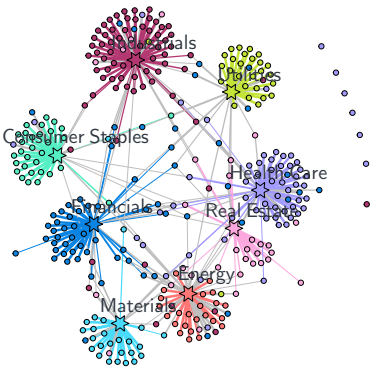


Heavy-tailed MRF graph

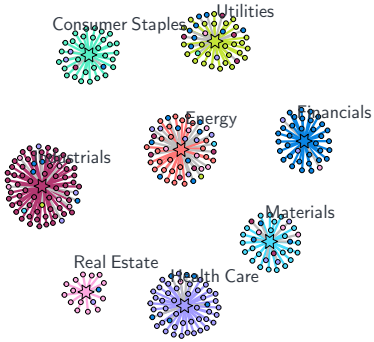
▶ *Heavy-tailed k-component graphs yield cleaner, more interpretable currency clusters than Gaussian counterparts.*

Numerical Experiments: Comparison of k-Component Bipartite Graphs

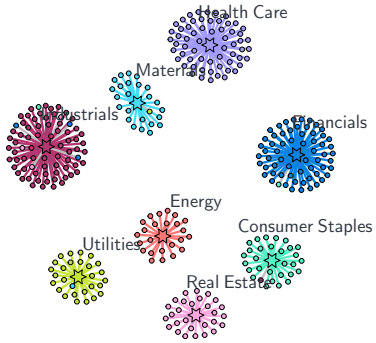
S&P 500 stocks (benchmarks: Kumar et al. 2020; Nie et al. 2017):



SGLA (Kumar et al., 2020)



SOBG (Nie et al., 2017)

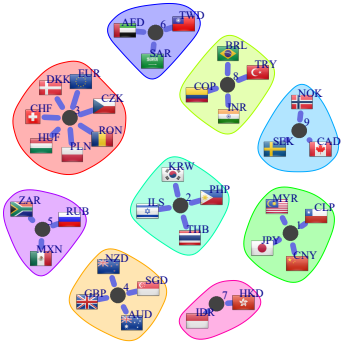


k-component bipartite heavy-tailed MRF

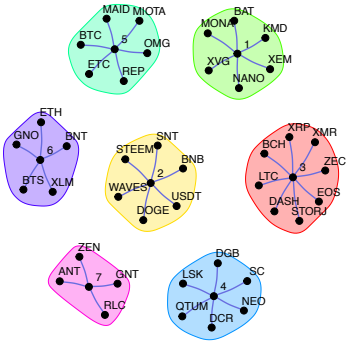
▶ *k-component bipartite heavy-tailed MRF graphs avoid isolated nodes and achieves better accuracy.*

Numerical Experiments: k-Component Bipartite Graphs with Unknown Indices

Applied to forex and crypto markets — unlike S&P 500, no predefined sector labels are available, so the k components and their cluster assignments are learned entirely from data (Javaheri and Palomar 2025).



Forex data



Crypto data

Outline

Motivation: Graphs Change Over Time

- Financial markets are **non-stationary**: correlations between assets shift during crises, regime changes, and macroeconomic events.

- **Rolling-window approach** (re-estimate graph on a sliding window of data):
  - Introduces estimation delay proportional to window size
  - Ignores temporal continuity — consecutive graphs are estimated independently
  - Not robust to outliers or missing observations

- **Principled time-varying formulation**: learn a sequence of graphs $\{\boldsymbol{L}_n\}$ jointly, coupling consecutive graphs through a temporal prior — capturing smooth evolution while adapting to regime shifts.

Time-Varying Graph Model

- Data arrives in **frames** $\mathcal{F}_n = \{t : t \in \text{frame } n\}$; each frame has its own graph $\mathcal{L}(\boldsymbol{w}_n)$ with edge weights $\boldsymbol{w}_n \in \mathbb{R}^{p(p-1)/2}$.
- The standard approach is to penalize the difference $\boldsymbol{w}_n - \boldsymbol{w}_{n-1}$ via the ℓ_2-norm or ℓ_1-norm.
- **Non-negative VAR(1) temporal evolution model**: edge weights evolve as

$$\boldsymbol{w}_n = (\boldsymbol{a} \odot \boldsymbol{w}_{n-1} + \varepsilon_n)_+, \quad n = 1, 2, \ldots$$

  where $\boldsymbol{a} \in \mathbb{R}_+^{p(p-1)/2}$ are VAR coefficients, ε_n is zero-mean noise, and $(\cdot)_+$ enforces non-negativity of edge weights.
- The **multiplicative factor** $\boldsymbol{a}$ better captures proportional changes in edge strengths over time.

(Javaheri et al. 2025)

Heavy Tails + Time-Varying: Combined Model

- **Laplacian operator** (Kumar et al. 2019, 2020): any non-negative edge weight vector $\boldsymbol{w} \in \mathbb{R}_+^{p(p-1)/2}$ defines a valid Laplacian via the linear mapping $\mathcal{L}(\boldsymbol{w})$: $\mathcal{L}(\boldsymbol{w})_{ij} = -w_{ij}$ for $i \neq j$ and $\mathcal{L}(\boldsymbol{w})_{ii} = \sum_{j \neq i} w_{ij}$. The optimization is then carried out directly over $\boldsymbol{w} \geq \mathbf{0}$.

- **Student-t likelihood** at each frame n combined with the VAR temporal prior yields (Javaheri et al. 2025):

$$\begin{aligned} \underset{\boldsymbol{w}_n \geq \mathbf{0},\, \boldsymbol{a} \geq \mathbf{0}}{\text{maximize}} \quad & \log \operatorname{gdet}(\mathcal{L}(\boldsymbol{w}_n)) - \frac{\nu + p}{T_n} \sum_{t \in \mathcal{F}_n} \log\left(1 + \frac{\boldsymbol{x}_t^\top \mathcal{L}(\boldsymbol{w}_n) \boldsymbol{x}_t}{\nu}\right) \\ & -\alpha \|\boldsymbol{w}_n - \boldsymbol{a} \odot \hat{\boldsymbol{w}}_{n-1}\|_1 - \beta \|\boldsymbol{w}_n\|_0 - \gamma \|\boldsymbol{a}\|_1 \\ \text{subject to} \quad & \boldsymbol{w}_n \in \Omega_w, \end{aligned}$$

where Ω_w encodes k-component structure and degree constraints.

Handling Corrupted and Missing Data

- In practice, financial data contains **missing observations and outlier-corrupted measurements**:

$$\boldsymbol{y}_t = \boldsymbol{m}_t \odot (\boldsymbol{x}_t + \boldsymbol{n}_t), \quad \boldsymbol{n}_t \sim \mathcal{N}(\boldsymbol{0}, \sigma_n^2 \boldsymbol{I}),$$

  where $\boldsymbol{m}_t \in \{0,1\}^p$ is a binary sampling mask ($m_{ti} = 0$ means asset i is missing at time t).

- The framework **jointly recovers the clean signal $\boldsymbol{x}_t$ and learns the graph $\boldsymbol{L}_{w_n}$** — missing entries are imputed as a byproduct.

- The MM weights $w_t^k \propto (\nu + \boldsymbol{x}_t^\top \boldsymbol{L}_{w_n} \boldsymbol{x}_t)^{-1}$ automatically **downweight outliers**: observations that are inconsistent with the graph structure receive lower weight.

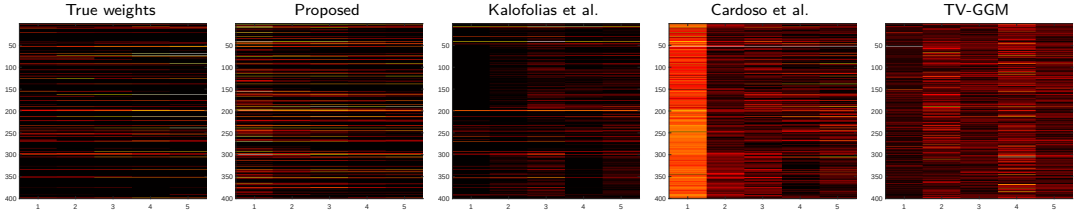
(Javaheri et al. 2025)

Algorithm: MM + ADMM

- The combined problem is non-convex. An **MM** outer loop linearizes the Student-t log terms, yielding a sequence of tractable subproblems solved via **ADMM**.
- Each iteration alternates five closed-form updates:
  - $\boldsymbol{L}_n$-update: eigendecomposition enforcing k-component rank constraint
  - $\boldsymbol{w}_n$-update: soft-thresholding enforcing sparsity (ℓ_0 relaxed to weighted ℓ_1)
  - $\boldsymbol{X}_n$-update: gradient step for signal recovery / missing-data imputation
  - $\boldsymbol{a}$-update: soft-thresholding for VAR coefficient learning
  - Dual variable update: standard ADMM multiplier ascent
- **Complexity:** $\mathcal{O}(p^3 + T_n p^2)$ per iteration; converges to a stationary point.
- The following slides show experiments on synthetic data and real S&P 500 market data (Javaheri et al. 2025).

Numerical Experiments: Synthetic Data

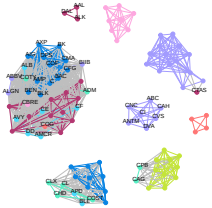
Recovered edge weights over time (darker = stronger edge):



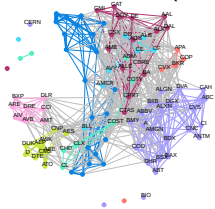
- *Proposed method recovers the ground-truth edge weight evolution most faithfully.*

Numerical Experiments: k-Component Graphs (S&P 500)

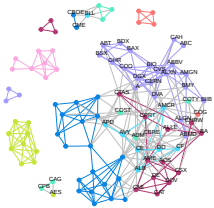
100 S&P 500 stocks partitioned into $k = 8$ sectors (last frame):



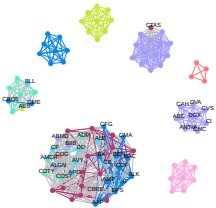
CLR (ACC 0.56)



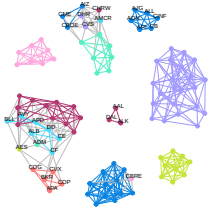
SGLA (ACC 0.27)



Fingraph (ACC 0.49)



Javaheri et al. (ACC 0.61)

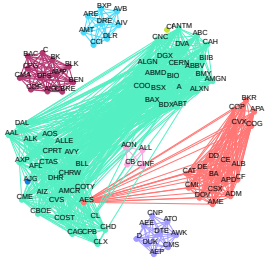


Proposed (ACC 0.69)

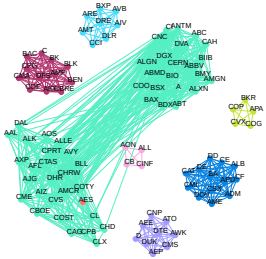
- *Proposed method cleanly separates sectors with fewer misclassified stocks.*

Numerical Experiments: Time Evolution of Graphs

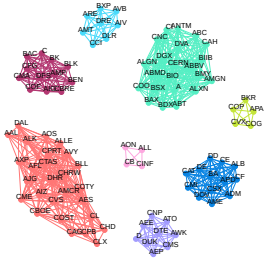
Graph structure evolves across three consecutive frames (S&P 500):



Frame 1



Frame 2

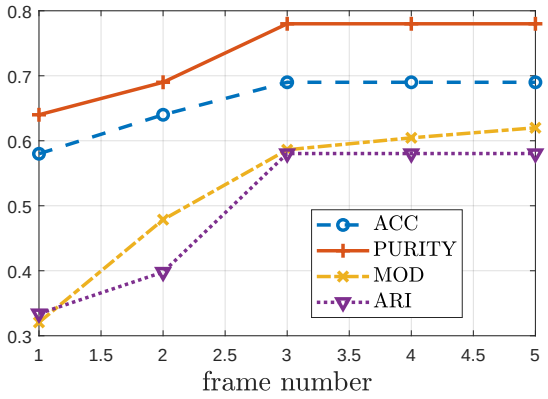


Frame 3

- *Cluster structure remains stable while edge strengths adapt to changing markets.*

Numerical Experiments: Clustering Accuracy Over Time

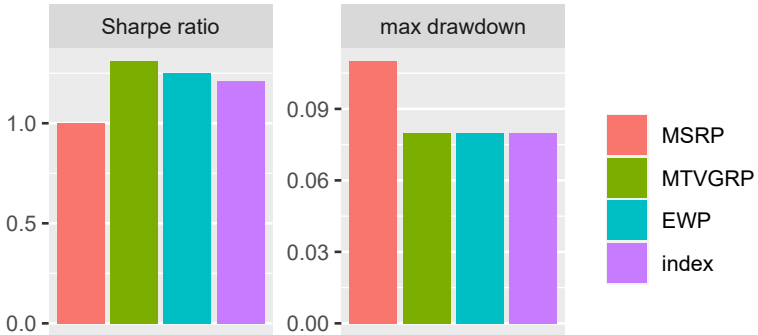
Clustering performance improves as more frames are observed (S&P 500, $k = 8$):



- *Time-varying model accumulates information across frames, steadily improving accuracy.*

Numerical Experiments: Portfolio Performance

Portfolio backtesting on S&P 500 (Sharpe ratio and maximum drawdown):



▶ *Time-varying graph portfolio (MTVGRP) achieves higher Sharpe ratio (1.31 vs. 1.00) and lower drawdown than benchmarks.*

Outline

Summary

- **Financial data is non-Gaussian and time-varying:** heavy tails, volatility clustering, and correlation structure all call for specialized graph models
- **Graph learning from smooth signals / GMRF** (Sections 2–3): a rich family of formulations — sparse, structured, and polynomial — for static graphs
- **Polynomial Graphical Lasso** (Section 3): exploits graph stationarity ($\boldsymbol{\Sigma S} = \boldsymbol{S\Sigma}$) to jointly learn the precision matrix and the graph, capturing multi-hop interactions (Buciulea et al. 2025)
- **Heavy-tailed and structured graphs** (Sections 5–6): Student-t model via MM yields robust estimation; spectral constraints enforce k-component and bipartite structure (M. Cardoso, Ying, and Palomar 2021, 2022; Kumar et al. 2020)
- **Time-varying graphs** (Section 7): non-negative VAR(1) temporal prior + Student-t likelihood tracks market dynamics, detects crises, and improves portfolio Sharpe ratio (Javaheri et al. 2025)

Thanks

For more information visit:

https://portfoliooptimizationbook.com
https://www.danielppalomar.com
https://github.com/convexfi
https://www.youtube.com/danielpalomar

References I

Buciulea, Andrei, Jiaxi Ying, Antonio G. Marques, and Daniel P. Palomar. 2025. "Polynomial Graphical Lasso: Learning Edges from Gaussian Graph-Stationary Signals." *IEEE Transactions on Signal Processing* 73: 1153–67. https://doi.org/10.1109/TSP.2025.3544376.

Cont, R. 2001. "Empirical Properties of Asset Returns: Stylised Facts and Statistical Issues." *Quantitative Finance* 1: 223–36.

Dong, X., D. Thanou, P. Frossard, and P. Vandergheynst. 2015. "Laplacian Matrix Learning for Smooth Graph Signal Representation." In *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*. Brisbane, Australia.

Dong, X., D. Thanou, M. Rabbat, and P. Frossard. 2019. "Learning Graphs from Data: A Signal Representation Perspective." *IEEE Signal Processing Magazine* 36 (3): 44–63.

References II

Egilmez, H. E., E. Pavez, and A. Ortega. 2017. "Graph Learning from Data Under Laplacian and Structural Constraints." *IEEE Journal of Selected Topics in Signal Processing* 11 (6): 825–41.

Javaheri, Amirhossein, and Daniel P. Palomar. 2025. "Clustering of Incomplete Data via a Bipartite Graph Structure." In *Proc. European Signal Processing Conference (EUSIPCO)*. Palermo, Italy.

Javaheri, Amirhossein, Jiaxi Ying, Daniel P. Palomar, and Farokh Marvasti. 2025. "Time-Varying Graph Learning for Data with Heavy-Tailed Distribution." *IEEE Transactions on Signal Processing* 73: 3044–60. https://doi.org/10.1109/TSP.2025.3588173.

Kalofolias, V. 2016. "How to Learn a Graph from Smooth Signals." In *Proc. Int. Conf. Artif. Intell. Statist.*, 920–29.

References III

Kumar, S., J. Ying, J. V. de M. Cardoso, and D. P. Palomar. 2019. "Structured Graph Learning via Laplacian Spectral Constraints." In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada.

———. 2020. "A Unified Framework for Structured Graph Learning via Spectral Constraints." *Journal of Machine Learning Research*, 1–60.

Lake, B., and J. Tenenbaum. 2010. "Discovering Structure by Learning Sparse Graphs." In *Proc. 33rd Annual Cognitive Science Conference*.

M. Cardoso, J. V. de, J. Ying, and D. P. Palomar. 2021. "Graphical Models for Heavy-Tailed Markets." In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*.

———. 2022. "Learning Bipartite Graphs: Heavy Tails and Multiple Components." In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*. New Orleans, LA, USA.

Mateos, G., S. Segarra, A. G. Marques, and A. Ribeiro. 2019. "Connecting the Dots." *IEEE Signal Processing Magazine* 36 (3): 16–43.

References IV

Meinshausen, N., and P. Bühlmann. 2006. "High-Dimensional Graphs and Variable Selection with the Lasso." *The Annals of Statistics* 34 (3): 1436–62.

Nie, F., X. Wang, M. Jordan, and H. Huang. 2016. "The Constrained Laplacian Rank Algorithm for Graph-Based Clustering." In *Proc. AAAI Conf. On Artificial Intelligence*, 1969–76.

Palomar, Daniel P. 2025. *Portfolio Optimization: Theory and Application*. Cambridge University Press.

Ruppert, D. 2010. *Statistics and Data Analysis for Financial Engineering*. Springer.

Sun, Y., P. Babu, and D. P. Palomar. 2017. "Majorization-Minimization Algorithms in Signal Processing, Communications, and Machine Learning." *IEEE Trans. Signal Processing* 65 (3): 794–816.

References V

Ying, J., J. V. de M. Cardoso, and D. P. Palomar. 2020. "Nonconvex Sparse Graph Learning Under Laplacian Constrained Graphical Model." In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*.

———. 2021. "Minimax Estimation of Laplacian Constrained Precision Matrices." In *Proc. Int. Conf. Artificial Intelligence and Statistics (AISTATS)*.

Zhao, L., Y. Wang, S. Kumar, and D. P. Palomar. 2019. "Optimization Algorithms for Graph Laplacian Estimation via ADMM and MM." *IEEE Trans. Signal Processing* 67 (16): 4231–44.