The Power Mean Laplacian for Multilayer Graph Clustering

Pedro Mercado\textsuperscript{1} \hspace{1cm} Antoine Gautier\textsuperscript{1} \hspace{1cm} Francesco Tudisco\textsuperscript{2} \hspace{1cm} Matthias Hein\textsuperscript{1}

\textsuperscript{1}Department of Mathematics and Computer Science, Saarland University, Germany
\textsuperscript{2}Department of Mathematics and Statistics, University of Strathclyde, G11XH Glasgow, UK

Abstract

Multilayer graphs encode different kind of interactions between the same set of entities. When one wants to cluster such a multilayer graph, the natural question arises how one should merge the information from different layers. We introduce in this paper a one-parameter family of matrix power means for merging the Laplacians from different layers and analyze it in expectation in the stochastic block model. We show that this family allows to recover ground truth clusters under different settings and verify this in real world data. While computing the matrix power mean can be very expensive for large graphs, we introduce a numerical scheme to efficiently compute its eigenvectors for the case of large sparse graphs.

1 Introduction

Multilayer graphs have received an increasing amount of attention due to their capability to encode different kinds of interactions between the same set of entities [6, 27]. This kind of graphs arise naturally in diverse applications such as transportation networks [16], financial-asset markets [4], temporal dynamics [53, 54], semantic world clustering [48], multimedia face analysis [7], mobile phone networks [26], social balance [8], citation analysis [52], and many others. The extension of clustering techniques to multilayer graphs is a challenging task and several approaches have been proposed so far. See [25, 51, 57, 63] for an overview. For instance, [13, 14, 52, 62] rely on matrix factorizations, whereas [11, 39, 41, 46, 47] take a Bayesian inference approach, and [28, 29] enforce consistency among layers in the resulting clustering assignment. In [37, 40, 55] Newman’s modularity [38] is extended to multilayer graphs. Recently [12, 50] proposed to compress a multilayer graph by combining sets of similar layers (called ‘strata’) to later identify the corresponding communities. Of particular interest to our work is the popular approach [1, 9, 23, 53, 64] that first blends the information of a multilayer graph by finding a suitable weighted arithmetic mean of the layers and then apply standard clustering methods to the resulting mono-layer graph.

In this paper we focus on extensions of spectral clustering to multilayer graphs. Spectral clustering is a well established method for one-layer graphs which, based on the first eigenvectors of the graph Laplacian, embeds nodes of the graphs in $\mathbb{R}^k$ and then uses $k$-means to find the partition. We propose to blend the information of a multilayer graph by taking certain matrix power means of Laplacians of the layers.

The power mean of scalars is a general family of means that includes as special cases, the arithmetic, geometric and harmonic means. The arithmetic mean of Laplacians has been used before in the case of signed networks [30] and thus our family of matrix power means, see Section 2.2, is a natural extension of this approach. One of our main contributions is to show that the arithmetic mean is actually suboptimal to merge information from different layers.

We analyze the family of matrix power means in the Stochastic Block Model (SBM) for multilayer graphs in two settings, see Section 3. In the first one all the layers are informative, whereas in the second setting none of the individual layers contains the full information but only if one considers them all together. We show that as the parameter of the family of Laplacian means tends to $-\infty$, in expectation one can recover perfectly the clusters in both situations. We provide extensive experiments which show that this behavior is stable when one samples sparse graphs from the SBM. Moreover, in Section 5, we provide additional experiments on real world graphs which confirm our finding in the SBM.
A main challenge for our approach is that the matrix power mean of sparse matrices is in general dense and thus does not scale to large sparse networks in a straightforward fashion. Thus a further contribution of this paper is to show that the first few eigenvectors of the matrix power mean can be computed efficiently. Our algorithm combines the power method with a Krylov subspace approximation technique and allows to compute the extremal eigenvalues and eigenvectors of the power mean of matrices without ever computing the matrix itself.

\section{Spectral clustering of multilayer graphs using matrix power means of Laplacians}

Let \( V = \{v_1, \ldots, v_n\} \) be a set of nodes and let \( T \) the number layers, represented by adjacency matrices \( W = \{W^{(1)}, \ldots, W^{(T)}\} \). For each non-negative weight matrix \( W^{(t)} \in \mathbb{R}_{+}^{n \times n} \) we have a graph \( G^{(t)} = (V, W^{(t)}) \) and a multilayer graph is the set \( G = \{G^{(1)}, \ldots, G^{(T)}\} \). In this paper our main focus are assortative graphs.

This kind of graphs are the most common in the literature (see f.i. [34]) and are used to model the situation where edges carry similarity information of pairs of vertices and thus are indicative for vertices being in the same cluster. For an assortative graph \( G = (V, W) \) spectral clustering is typically based on the Laplacian matrix and its normalized version, defined respectively as

\[
L = D - W \quad L_{\text{sym}} = D^{-1/2}LD^{-1/2}
\]

where \( D_{ii} = \sum_{j=1}^{n} w_{ij} \) is the diagonal matrix of the degrees of \( G \). Both Laplacians are symmetric positive semi-definite and the multiplicity of eigenvalue 0 is equal to the number of connected components in \( G \).

Given a multilayer graph with all assortative layers \( G^{(1)}, \ldots, G^{(T)} \), our goal is to come up with a clustering of the vertex set \( V \). We point out that in this paper a clustering is a partition of \( V \), that is each vertex is uniquely assigned to one cluster.

\subsection{Matrix power mean of Laplacians for multilayer graphs}

Let us briefly recall the scalar power mean of a set of non-negative scalars \( x_1, \ldots, x_T \). This is a general one-parameter family of means defined for \( p \in \mathbb{R} \) as

\[
m_p(x_1, \ldots, x_T) = \left( \frac{1}{T} \sum_{i=1}^{T} x_i^p \right)^{1/p}.
\]

It includes some well-known means as special cases:

\[
\lim_{p \to \infty} m_p(x_1, \ldots, x_T) = \max\{x_1, \ldots, x_T\}
\]

\[
\lim_{p \to 0} m_p(x_1, \ldots, x_T) = \frac{x_1 + \cdots + x_T}{T}
\]

\[
\lim_{p \to -\infty} m_p(x_1, \ldots, x_T) = \sqrt[2]{\frac{x_1 \cdot \cdots \cdot x_T}
\]

\[
\lim_{p \to 1} m_p(x_1, \ldots, x_T) = \min\{x_1, \ldots, x_T\}
\]

corresponding to the maximum, arithmetic, geometric, harmonic mean and minimum, respectively.

Since matrices do not commute, the scalar power mean can be extended to positive definite matrices in a number of different ways, all of them coinciding when applied to commuting matrices. In this work we use the following matrix power mean.

\begin{definition}[5] Let \( A_1, \ldots, A_T \) be symmetric positive definite matrices, and \( p \in \mathbb{R} \). The matrix power mean of \( A_1, \ldots, A_T \) with exponent \( p \) is

\[
M_p(A_1, \ldots, A_T) = \left( \frac{1}{T} \sum_{i=1}^{T} A_i^p \right)^{1/p}
\]

where \( A^{1/p} \) is the unique positive definite solution of the matrix equation \( X^p = A \).
\end{definition}

The previous definition can be extended to positive semi-definite matrices. For \( p > 0 \), \( M_p(A_1, \ldots, A_T) \) exists for positive semi-definite matrices, whereas for \( p \leq 0 \) it is necessary to add a suitable diagonal shift to \( A_1, \ldots, A_T \) to enforce them to be positive definite (see [5] for details).

We call the matrix above matrix power mean and we recover for \( p = 1 \) the standard arithmetic mean of the matrices. Note that for \( p \to 0 \), the power mean (1) converges to the Log-Euclidean matrix mean [3]

\[
M_0(A_1, \ldots, A_T) = \exp \left( \frac{1}{T} \sum_{i=1}^{T} \log A_i \right)
\]

which is a popular form of matrix geometric mean used, for instance, in diffusion tensor imaging or quantum information theory (see f.i. [2, 42]).

Based on the Karcher mean, a different one-parameter family of matrix power means has been discussed for instance in [31]. When the parameter goes to zero, the Karcher-based power mean of two matrices \( A \) and \( B \) converges to the geometric mean \( A \# B = A^{1/2}(A^{-1/2}BA^{-1/2})^{1/2}A^{-1/2} \). The mean \( A \# B \) has been used for instance in [15, 36] for clustering in signed networks, for metric learning [61] and geometric optimization [49]. However, when more than two matrices are considered, the Karcher-based power mean is defined as the solution of a set of nonlinear matrix equations with no known closed-form solution and thus is not suitable for multilayer graphs.
The matrix power mean (1) is symmetric positive definite and is independent of the labeling of the vertices in the sense that the matrix power mean of relabeled matrices is the same as relabeling the matrix power mean of the original matrices. The latter property is a necessary requirement for any clustering method. The following lemma illustrates the relation to the scalar power mean and is frequently used in the proofs.

**Lemma 1.** Let $u$ be an eigenvector of $A_1, \ldots, A_T$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_T$. Then $u$ is an eigenvector of $M_p(A_1, \ldots, A_T)$ with eigenvalue $m_p(\lambda_1, \ldots, \lambda_T)$.

### 2.2 Matrix power means for multilayer spectral clustering

We consider the multilayer graph $G = (G^{(1)}, \ldots, G^{(T)})$ and define the **power mean Laplacian** $L_p$ of $G$ as

$$L_p = M_p(L_{sym}^{(1)}, \ldots, L_{sym}^{(T)})$$

where $L_{sym}^{(t)}$ is the normalized Laplacian of the graph $G^{(t)}$. Note that Definition 1 of the matrix power mean $M_p(A_1, \ldots, A_T)$ requires $A_1, \ldots, A_T$ to be positive definite. As the normalized Laplacian is positive semidefinite, in the following, for $p \leq 0$ we add to $L_{sym}^{(t)}$ in Equation (2) a small diagonal shift which ensures positive definiteness, that is we consider $L_{sym}^{(t)} + \varepsilon I$ throughout the paper. For all numerical experiments we set $\varepsilon = \log(1 + |p|)$ for $p < 0$ and $\varepsilon = 10^{-6}$ for $p = 0$. Abusing notation slightly, we always mean the shifted versions in the following, unless the shift is explicitly stated.

Similar to spectral clustering for a single graph, we propose Alg. 1 for the spectral clustering of multilayer graphs based on the matrix power mean of Laplacians. As in standard spectral clustering, see [34], our algorithm 1 uses the eigenvectors corresponding to the $k$ smallest eigenvalues of the power mean Laplacian $L_p$. Thus the relative ordering of the eigenvalues of $L_p$ is of utmost importance. By Lemma 1 we know that if $A_i u = \lambda(A_i) u$, for $i = 1, \ldots, n$, then the corresponding eigenvalue of the matrix power mean is $m_p(\lambda(A_1), \ldots, \lambda(A_T))$. Hence, the ordering of eigenvalues strongly depends on the choice of the parameter $p$. In the next Section we study the effect of the parameter $p$ on the ordering of the eigenvectors of $L_p$ for multilayer graphs following the stochastic block model.

### 3 Stochastic block model on multilayer graphs

In this Section we present an analysis of the eigenvectors and eigenvalues of the power mean Laplacian under the Stochastic Block Model (SBM) for multilayer graphs. The SBM is a widespread random graph model for single-layer networks having a prescribed clustering structure [44]. Studies of community detection for multilayer networks following the SBM can be found in [19, 21, 24, 58, 59, 60].

In order to grasp how different methods identify communities in multilayer graphs following the SBM we will analyze three different settings. In the first setting all layers follow the same node partition (see f.i. [19]) and we study the robustness of the spectrum of the power mean Laplacian when the first layer is informative and the other layers are noise or even contain contradicting information. In the second setting we consider the particularly interesting situation where multilayer-clustering is superior over each individual clustering. More specifically, we consider the case where we are searching for three clusters but each layer contains only information about one of them and only considering all of the layers together reveals the information about the underlying cluster structure. In a third setting we go beyond the standard SBM and consider the case where we have a graph partition for each layer, but this partition changes from layer to layer according to a generative model (see f.i. [4]). However, for the last setting we only provide an empirical study, whereas for the first two settings we analyze the spectrum also analytically. For brevity, all the proofs are moved to the supplementary material.

In the following we denote by $D_1, \ldots, D_k$ the ground truth clusters that we aim to recover. All the $C_i$ are assumed to have the same size $|C|$. Calligraphic letters are used for the expected matrices in the SBM. In particular, for a layer $G^{(t)}$ we denote by $W^{(t)}$ its expected adjacency matrix, by $D^{(t)} = \text{diag}(W^{(t)1})$ the expected degree matrix and by $L_{sym}^{(t)} = I - (D^{(t)})^{-1/2}W^{(t)}(D^{(t)})^{-1/2}$ the expected normalized Laplacian.

#### 3.1 Case 1: Robustness to noise where all layers have the same cluster structure

The case where all layers follow a given node partition
is a natural extension of the mono-layer SBM to the multilayer setting. This is done by having different edge probabilities for each layer [19], while fixing the same node partition in all layers. We denote by $p_{in}^{(t)}$ (resp. $p_{out}^{(t)}$) the probability that there exists an edge in layer $G^{(t)}$ between nodes that belong to the same (resp. different) clusters. Then $W_{ij}^{(t)} = p_{in}^{(t)}$ if $v_i, v_j$ belong to the same cluster and $W_{ij}^{(t)} = p_{out}^{(t)}$ if $v_i, v_j$ belong to different clusters. Consider the following $k$ vectors:

$$X_1 = 1, \quad X_k = (k-1)1_{C_k} - 1_{P_k}.$$

The use of $k$-means on the embedding induced by the vectors $\{X_i\}_{i=1}^{k}$ identifies the ground truth communities $\{C_i\}_{i=1}^{k}$. It turns out that in expectation $\{X_i\}_{i=1}^{k}$ are eigenvectors of the power mean Laplacian $L_p$. We look for conditions so that they correspond to the $k$ smallest eigenvalues as this implies that our spectral clustering Algorithm 1 recovers the ground truth.

Before addressing the general case, we discuss the case of two layers. For this case we want to illustrate the effect of the power mean by simply studying the extreme limit cases

$$L_\infty := \lim_{p \rightarrow \infty} L_p \quad \text{and} \quad L_- := \lim_{p \rightarrow -\infty} L_p.$$

where $L_p = M_p(L_{sym}, L_{sym})$. The next Lemma shows that $L_- \approx L_\infty$ are related to the logical operators AND and OR, respectively, in the sense that in expectation $L_- \approx L_{\infty}$ recovers the clusters if and only if $G^{(1)}$ and $G^{(2)}$ have both clustering structure, whereas in expectation $L_- \approx L_{\infty}$ recovers the clusters if and only if $G^{(1)}$ or $G^{(2)}$ has clustering structure.

Lemma 2. Let $L_p = M_p(L_{sym}, L_{sym})$:

1. $\{X_i\}_{i=1}^{k}$ correspond to the $k$ smallest eigenvalues of $L_\infty$ if and only if $p_{in}^{(1)} > p_{out}^{(1)}$ and $p_{in}^{(2)} > p_{out}^{(2)}$.
2. $\{X_i\}_{i=1}^{k}$ correspond to the $k$ smallest eigenvalues of $L_- \approx L_{\infty}$ if and only if $p_{in}^{(1)} > p_{out}^{(1)}$ or $p_{in}^{(2)} > p_{out}^{(2)}$.

The following theorem gives general conditions on the recovery of the ground truth clusters in dependency on $p$ and the size of the shift in $L_p$, see Section 2.2. Note that, in analogy with Lemma 2, as $p \rightarrow -\infty$ the recovery of the ground truth clusters is achieved if at least one of the layers is informative, whereas if $p \rightarrow \infty$ all of them have to be informative in order to recover the ground truth.

Theorem 1. Let $p \in [-\infty, \infty]$, then $X_1, \ldots, X_k$ correspond to the $k$-smallest eigenvalues of $L_p$ if and only if $m_p(\mu + 1) < 1 + \epsilon$, where $\mu = (1 - p_1, \ldots, 1 - \rho_T)$, and $\rho_t = (p_{in}^{(t)} - p_{out}^{(t)})/(p_{in}^{(t)} - (k-1)p_{out}^{(t)})$.

In particular, for $p \rightarrow \pm \infty$, we have

1. $X_1, \ldots, X_k$ correspond to the $k$-smallest eigenvalues of $L_\infty$ if and only if all layers are informative, i.e. $p_{in}^{(t)} > p_{out}^{(t)}$ holds for all $t \in \{1, \ldots, T\}$.

Theorem 1 shows that the informative eigenvectors of $L_p$ are at the bottom of the spectrum if and only if the scalar power mean of the corresponding eigenvalues is small enough. Since the scalar power mean is monotonically decreasing with respect to $p$, this explains why the limit case $p \rightarrow \infty$ is more restrictive than $p \rightarrow -\infty$. The corollary below shows that the coverage of parameter settings in the SBM for which one recovers the ground truth becomes smaller as $p$ grows.

Corollary 1. Let $q \leq p$. If $X_1, \ldots, X_k$ correspond to the $k$-smallest eigenvalues of $L_p$, then $X_1, \ldots, X_k$ correspond to the $k$-smallest eigenvalues of $L_q$.

The previous results hold in expectation. The following experiments show that these findings generalize to the case where one samples from the SBM. In Fig. 1 we present experiments on sparse sampled multilayer graphs from the SBM. We consider two clusters of size $|C| = 100$ and show the mean of clustering error of 50 runs. We evaluate the power mean Laplacian $L_p$ with $p \in \{0, \pm 1, \pm 2, \pm 5, \pm 10\}$ and compare with other methods described in Section 5.

In Fig. 1 we fix the first layer $G^{(1)}$ to be strongly assortative and let the second layer $G^{(2)}$ run from a disassortative to an assortative configuration. In Fig. 1a we
can see that the power mean Laplacian $L_{-10}$ returns the smallest clustering error, together with the multilateral method, the best single view and the heuristic approach across all parameter settings. The latter two work well by construction in this setting. However, we will see that they fail for the second setting we consider next. All the other competing methods fail as the layers, that is $p_{in} > p_{out}$. In this case spectral clustering applied on a single layer $W^{(t)}$ would return cluster $C_{t}$ and a random partition of the complement, failing to recover the ground truth clustering $C_{1}, C_{2}, C_{3}$. This is shown in the following Theorem.

**Theorem 2.** If $p_{in} > p_{out}$, then for any $t = 1, 2, 3$, there exist scalars $\alpha > 0$ and $\beta > 0$ such that the eigenvectors of $L_{\text{sym}}^{(t)}$ corresponding to the two smallest eigenvalues are

$$\chi_{1} = \alpha \mathbf{1}_{C_{t}} + 1_{\overline{C_{t}}} \quad \text{and} \quad \chi_{2} = -\beta \mathbf{1}_{C_{t}} + 1_{\overline{C_{t}}}$$

whereas any vector orthogonal to both $\chi_{1}$ and $\chi_{2}$ is an eigenvector for the third smallest eigenvalue.

On the other hand, it turns out that the power mean Laplacian $L_{p}$ is able to merge the information of each layer, obtaining the ground truth clustering, for all integer powers different from zero. This is formally stated in the following.

**Theorem 3.** Let $p_{in} > p_{out}$ and for $\varepsilon > 0$ define

$$\tilde{L}_{\text{sym}}^{(t)} = L_{\text{sym}}^{(t)} + \varepsilon I, \quad t = 1, 2, 3.$$  

Then the eigenvectors of $L_{p}(\tilde{L}_{\text{sym}}^{(1)}, \tilde{L}_{\text{sym}}^{(2)}, \tilde{L}_{\text{sym}}^{(3)})$ corresponding to its three smallest eigenvalues are

$$\chi_{1} = \mathbf{1}, \quad \chi_{2} = \mathbf{1}_{C_{t}} - \mathbf{1}_{\overline{C_{t}}} \quad \text{and} \quad \chi_{3} = \mathbf{1}_{\overline{C_{t}}} - \mathbf{1}_{C_{t}}$$

for any nonzero integer $p$.

The proof of Theorem 3 is more delicate than the one of Theorem 1, as it involves the addition of powers of matrices that do not have the same eigenvectors.
Note that Theorem 3 does not distinguish the behavior for distinct values of $p$. In expectation all nonzero integer values of $p$ work the same. This is different to Theorem 1, where the choice of $p$ had a relevant influence on the eigenvector embedding even in expectation. However, we see in the experiments on graphs sampled from the SBM (Figure 2) that the choice of $p$ has indeed a significant influence on the performance even though they are the same in expectation. This suggests that the smaller $p$, the smaller the variance in the difference to the expected behavior in the SBM. We leave this as an open problem if such a dependency can be shown analytically.

In Figs. 2a and 2b we present the mean clustering error out of ten runs. In Fig. 2a one can see that BestView and Heuristic, which rely on clusterings determined by single views, return high clustering errors which correspond to the identification of only a single cluster. The result of Theorem 3 explains this failure. The reason for the increasing clustering error with $p$ can be seen in Fig. 2c where we analyze how the ordering of eigenvectors changes for different values of $p$. We can see that for negative powers, the informative eigenvectors belong to the bottom three eigenvalues (denoted in red). For the cases where $p \geq 2$ the ordering changes, pushing non-informative eigenvectors to the bottom of the spectrum and thus resulting into a high clustering error, as seen in Fig. 2d. However, we conclude that also for this second case a strongly negative power mean Laplacian as $L^{-10}$ works best.

### 3.3 Case 3: Non-consistent partitions between layers

We now consider the case where all the layers follow the same node partition (as in Section 3.1), but the partitions may fluctuate from layer to layer with a certain probability. We use the multilayer network model introduced in [4]. This generative model considers a graph partition for each layer, allowing the partitions to change from layer to layer according to an interlayer dependency tensor. For the sake of clarity we consider a one-parameter interlayer dependency tensor with parameter $\tilde{p} \in [0, 1]$ (i.e. a uniform multiplex network according to the notation used in Section 3B in [4]), where for $\tilde{p} = 0$ the partitions between layers are independent, and for $\tilde{p} = 1$ the partitions between layers are identical. Once the partitions are obtained, edges are generated following a multilayer degree-corrected SBM (DCSBM in Section 4 of [4]), according to a one-parameter affinity matrix with parameter $\mu \in [0, 1]$, where for $\mu = 0$ all edges are within communities whereas for $\mu = 1$ edges are assigned ignoring the community structure.

We choose $\tilde{p} \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ and $\mu \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ and consider all possible combinations of $(\tilde{p}, \mu)$. For each pair we count how many times, out of 50 runs, each method achieves the smallest clustering error. The remaining parameters of the DCSBM are set as follows: exponent $\gamma = -3$, minimum degree and maximum degree $k_{\text{min}} = k_{\text{max}} = 10$, $|V| = 100$ nodes, $T = 10$ layers and $K = 2$ communities. As partitions between layers are not necessarily the same, we take the most frequent node assignment among all 10 layers as ground truth clustering.

In Table 1, left side, we show the result for fixed values of $\tilde{p}$ and average over all values of $\mu$. On the right table we show the corresponding results for fixed values of $\mu$ and average over all values of $\tilde{p}$. On the left table we can see that for $\tilde{p} = 1$, where the partition is the same in all layers, all methods recover the clustering, while, as one would expect, the performance decreases with smaller values of $\tilde{p}$. Further, we note that the per-

<table>
<thead>
<tr>
<th>$\tilde{p}$</th>
<th>$L_{agg}$</th>
<th>Coreg</th>
<th>BestView</th>
<th>Heuristic</th>
<th>TLMV</th>
<th>RMSC</th>
<th>MT</th>
<th>$L_{10}$</th>
<th>$L_5$</th>
<th>$L_2$</th>
<th>$L_1$</th>
<th>$L_0$</th>
<th>$L_{-1}$</th>
<th>$L_{-2}$</th>
<th>$L_{-5}$</th>
<th>$L_{-10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.3</td>
<td>1.3</td>
<td>3.0</td>
<td>8.0</td>
<td>22.3</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.6</td>
<td>0.3</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>64.7</td>
<td>16.7</td>
<td>17.0</td>
<td>77.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>59.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.8</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mu$</th>
<th>$L_{agg}$</th>
<th>Coreg</th>
<th>BestView</th>
<th>Heuristic</th>
<th>TLMV</th>
<th>RMSC</th>
<th>MT</th>
<th>$L_{10}$</th>
<th>$L_5$</th>
<th>$L_2$</th>
<th>$L_1$</th>
<th>$L_0$</th>
<th>$L_{-1}$</th>
<th>$L_{-2}$</th>
<th>$L_{-5}$</th>
<th>$L_{-10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>24.7</td>
<td>21.7</td>
<td>21.7</td>
<td>21.7</td>
<td>24.3</td>
<td>21.3</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
</tr>
<tr>
<td>0.1</td>
<td>16.7</td>
<td>16.7</td>
<td>13.3</td>
<td>11.7</td>
<td>6.0</td>
<td>1.0</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
</tr>
<tr>
<td>0.2</td>
<td>16.7</td>
<td>17.0</td>
<td>17.0</td>
<td>17.7</td>
<td>11.7</td>
<td>9.0</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
<td>16.7</td>
</tr>
<tr>
<td>0.3</td>
<td>25.7</td>
<td>24.3</td>
<td>21.7</td>
<td>23.3</td>
<td>21.0</td>
<td>20.0</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
<td>25.7</td>
</tr>
<tr>
<td>0.4</td>
<td>26.3</td>
<td>22.0</td>
<td>23.0</td>
<td>21.7</td>
<td>20.3</td>
<td>20.0</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
<td>26.3</td>
</tr>
<tr>
<td>0.5</td>
<td>19.7</td>
<td>19.7</td>
<td>21.0</td>
<td>20.7</td>
<td>20.7</td>
<td>20.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
<td>19.7</td>
</tr>
</tbody>
</table>

Table 1: Percentage of cases where the minimum clustering error is achieved by different methods. Left: Columns correspond to a fixed value of $\tilde{p}$ and we aggregate over $\mu \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. Left: Columns correspond to a fixed value of $\mu$ and we aggregate over $\tilde{p} \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. 

This table shows the percentage of times each method achieved the lowest clustering error, out of 50 runs, for different values of $\tilde{p}$ and $\mu$. The table is divided into two sections, one for fixed $\tilde{p}$ and the other for fixed $\mu$. The columns represent different values of $\tilde{p}$ or $\mu$, while the rows correspond to different methods: Coreg, BestView, and Heuristic. The values indicate the percentage of times each method achieved the lowest clustering error.
formance of the power mean Laplacian improves as $\lambda$ decreases and $L_{-10}$ again achieves the best result. On the right table we see that performance is degrading with larger values of $\mu$. This is expected as for larger values of $\mu$ the edges inside the clusters are less concentrated. Again the performance of the power mean Laplacian improves as $p$ decreases and $L_{-10}$ performs best.

4 Computing the smallest eigenvalues and eigenvectors of $M_p(A_1, \ldots, A_T)$

We present an efficient method for the computation of the smallest eigenvalues of $M_p(A_1, \ldots, A_T)$ which does not require the computation of the matrix $M_p(A_1, \ldots, A_T)$. This is particularly important when dealing with large-scale problems as $M_p(A_1, \ldots, A_T)$ is typically dense even though each $A_i$ is a sparse matrix. We restrict our attention to the case $p < 0$ which is the most interesting one in practice. The positive case $p > 0$ as well as the limit case $p \to 0$ deserve a different analysis and are not considered here.

Let $A_1, \ldots, A_T$ be positive definite matrices. If $\lambda_1 \leq \cdots \leq \lambda_n$ are the eigenvalues of $M_p(A_1, \ldots, A_T)$ corresponding to the eigenvectors $u_1, \ldots, u_n$, then $\mu_i = (\lambda_i)^p$, $i = 1, \ldots, n$, are the eigenvalues of $M_p(A_1, \ldots, A_T)^p$ corresponding to the eigenvectors $u_i$. However, the function $f(x) = x^p$ is order reversing for $p < 0$. Thus, the relative ordering of the $\mu_i$’s changes into $\mu_1 \geq \cdots \geq \mu_n$. Thus, the smallest eigenvalues and eigenvectors of $M_p(A_1, \ldots, A_T)$ can be computed by addressing the largest ones of $M_p(A_1, \ldots, A_T)^p$. To this end we propose a power method type outer-scheme, combined with a Krylov subspace approximation inner-method. The pseudo code is presented in Alg. 2 and 3. Each step of the outer iteration in Alg. 2 requires to compute the $p$th power of $T$ matrices times a vector. Computing $A^p \times \text{vector}$, reduces to the problem of computing the product of a matrix function times a vector. Krylov methods are among the most efficient and most studied strategies to address such a computational issue. As $A^p$ is a polynomial in $A$, we apply a Polynomial Krylov Subspace Method (PKSM), whose pseudo code is presented in Alg. 3 and which we briefly describe in the following. For further details we refer to [22] and the references therein. For the sake of generality, below we describe the method for a general positive definite matrix $A$.

The general idea of PKSM $s$-th iteration is to project $A$ onto the subspace $K^s(A, y) = \text{span}\{y, Ay, \ldots, A^{s-1}y\}$ and solve the problem there. The projection onto $K^s(A, y)$ is realized by means of the Lanczos process, producing a sequence of matrices $V_s$ with orthogonal columns, where the first column of $V_s$ is $y/\|y\|_2$ and range($V_s$) = $K^s(A, y)$. Moreover at each step we have $AV_s = V_sH_s + v_{s+1}e_i^T$ where $H_s$ is $s \times s$ symmetric tridiagonal, and $e_i$ is the $i$-th canonical vector. The matrix vector product $x = A^py$ is then approximated by $x_s = V_sH_se_1\|y\|_2 \approx A^py$.

Clearly, if operations are done with infinite precision,
the exact $x$ is obtained after $n$ steps. However, in practice, the error $\|x_n - x\|$ decreases very fast with $s$ and often very few steps are enough to reach a desirable tolerance. Two relevant observations are in order: first, the matrix $H_s = V_s^T A V_s$ can be computed iteratively alongside the Lanczos method, thus it does not require any additional matrix multiplication; second, the $p$ power of the matrix $H_s$ can be computed directly without any notable increment in the algorithm cost, since $H_s$ is tridiagonal of size $s \times s$.

Several eigenvectors can be simultaneously computed with Algs. 2 and 3 by orthonormalizing the current eigenvector approximation at every step of the power method (Alg. 2) (see f.i. algorithm 5.1 Subspace iteration in [45]). Moreover, the outer iteration in Alg. 2 can be easily run in parallel as the vectors $u_{k}^{(i)}$, $i = 1, \ldots, T$ can be built independently of each other.

A numerical evaluation of Algs. 2 and 3 is presented in Fig. 3. We consider graphs of sizes $|V| \in \{1 \times 10^4, 2 \times 10^4, 3 \times 10^4, 4 \times 10^4\}$. Further, for each multilayer graph we generate two assortative graphs with parameters $p_{in} = 0.65$ and $p_{in} = 0.25$, following the SBM. Moreover, we consider the power mean Laplacian $L_p = M_p(L_{sym}^{(1)}, L_{sym}^{(2)})$ with parameter $p \in \{-1, -2, -5, -10\}$. As a baseline we take the arithmetic mean Laplacian $L_1 = M_1(L_{sym}^{(1)}, L_{sym}^{(2)})$ and use Matlab’s \texttt{eigs} function. For all cases, we compute the two eigenvectors corresponding to the smallest eigenvalues. We present the mean execution time of 10 runs. Experiments are performed using one thread.

<table>
<thead>
<tr>
<th></th>
<th>3Sources</th>
<th>BBC</th>
<th>BBCS</th>
<th>Wiki</th>
<th>UCI</th>
<th>Citeseer</th>
<th>Cora</th>
<th>WebKB</th>
</tr>
</thead>
<tbody>
<tr>
<td># vertices</td>
<td>169</td>
<td>685</td>
<td>544</td>
<td>693</td>
<td>2000</td>
<td>3312</td>
<td>2708</td>
<td>187</td>
</tr>
<tr>
<td># layers</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td># classes</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>10</td>
<td>10</td>
<td>6</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>$L_{agg}$</td>
<td>0.194</td>
<td>0.156</td>
<td>0.152</td>
<td>0.371</td>
<td>0.162</td>
<td>0.373</td>
<td>0.452</td>
<td>0.277</td>
</tr>
<tr>
<td>Coreg</td>
<td>0.215</td>
<td>0.196</td>
<td>0.164</td>
<td>0.784</td>
<td>0.248</td>
<td>0.395</td>
<td>0.659</td>
<td>0.444</td>
</tr>
<tr>
<td>Heuristic</td>
<td>0.192</td>
<td>0.218</td>
<td>0.198</td>
<td>0.697</td>
<td>0.280</td>
<td>0.474</td>
<td>0.515</td>
<td>0.400</td>
</tr>
<tr>
<td>TLMV</td>
<td>0.284</td>
<td>0.259</td>
<td>0.317</td>
<td>0.412</td>
<td>0.154</td>
<td>0.363</td>
<td>0.533</td>
<td>0.430</td>
</tr>
<tr>
<td>RMSC</td>
<td>0.254</td>
<td>0.255</td>
<td>0.194</td>
<td>0.407</td>
<td>0.173</td>
<td>0.422</td>
<td>0.507</td>
<td>0.279</td>
</tr>
<tr>
<td>MT</td>
<td>0.249</td>
<td>0.133</td>
<td>0.158</td>
<td>0.544</td>
<td>0.103</td>
<td>0.371</td>
<td>0.436</td>
<td>0.298</td>
</tr>
<tr>
<td>$L_1$</td>
<td>0.194</td>
<td>0.154</td>
<td>0.148</td>
<td>0.373</td>
<td>0.163</td>
<td>0.285</td>
<td>0.367</td>
<td>0.440</td>
</tr>
<tr>
<td>$L_{-10}$ (ours)</td>
<td>0.200</td>
<td>0.159</td>
<td>0.144</td>
<td>0.368</td>
<td>0.095</td>
<td>0.283</td>
<td>0.374</td>
<td>0.439</td>
</tr>
</tbody>
</table>

Table 2: Average Clustering Error

5 Experiments

We take the following baseline approaches of spectral clustering applied to: the average adjacency matrix ($L_{agg}$), the arithmetic mean Laplacian ($L_1$), the layer with the largest spectral gap (\textbf{Heuristic}), and to the layer with the smallest clustering error (\textbf{BestView}). Further, we consider: Pairwise Co-Regularized Spectral Clustering [29], with parameter $\lambda = 0.01$ (\textbf{Coreg}), which proposes a spectral embedding generating a clustering consistent among all graph layers, Robust Multi-View Spectral Clustering [56], with parameter $\lambda = 0.005$ (\textbf{RMSC}), which obtains a robust consensus representation by fusing noiseless information present among layers, spectral clustering applied to a suitable convex combination of normalized adjacency matrices [64] (\textbf{TLMV}), and a tensor factorization method [11] (\textbf{MT}), which considers a multi-layer mixed membership (SBM).

We take several datasets: \textit{3Sources}[32], BBC[17] and BBC Sports[18] news articles, a dataset of Wikipedia articles[43], the hand written UCI digits dataset with six different features and citations datasets \textit{Citeseer}[33], \textit{Cora}[35] and \textit{WebKB}[10], (from WebKB we only take the subset Texas). For each layer we build the corresponding adjacency matrix from the $k$-nearest neighbour graph based on the Pearson linear correlation between nodes, i.e. the higher the correlation the nearer the nodes are. We test all clustering methods over all choices of $k \in \{20, 40, 60, 80, 100\}$, and present the average clustering error in Table 2. Datasets \textit{CiteSeer}, \textit{Cora} and \textit{WebKB} have two layers: one is a fixed citation network, whereas the second one is the $k$-nearest neighbour graph built on documents features. We can see that in four out of eight datasets the power mean Laplacian $L_{-10}$ gets the smallest clustering error. The largest difference in clustering error is present in the UCI dataset, where the second best is MT. Further, $L_1$ presents the smallest clustering error in Cora, being $L_{-10}$ close to it. The smallest clustering error in WebKB is achieved by $L_{agg}$. This dataset is particularly challenging, due to conflictive layers[20].

Acknowledgments. The work of P.M., A.G. and M.H. has been funded by the ERC starting grant NOLEPRO n. 307793. The work of F.T. has been funded by the Marie Curie Individual Fellowship MAGNET n. 744014.
References


The Power Mean Laplacian for Multilayer Graph Clustering


