European Journal of Applied Mathematics (2026), **37**, pp. 743–825
doi:10.1017/S095679252400072X

CAMBRIDGE UNIVERSITY PRESS

PAPER

An MBO method for modularity optimisation based on total variation and signless total variation

Zijun Li[1], Yves van Gennip[2] and Volker John[3,4]

[1]Department of Mathematics, Humboldt-Universität zu Berlin, Berlin, Germany
[2]Delft Institute of Applied Mathematics, Delft University of Technology, Delft, Netherlands
[3]Weierstrass Institute for Applied Analysis and Stochastics, Berlin, Germany
[4]Department of Mathematics and Computer Science, Freie Universität Berlin, Berlin, Germany
Corresponding author: van Gennip Yves; Email: y.vangennip@tudelft.nl

Received: 31 January 2024; **Revised:** 09 August 2024; **Accepted:** 12 September 2024;
First published online: 25 November 2024

Keywords: modularity; MBO scheme; Ginzburg–Landau functional; community detection; data clustering

2020 Mathematics Subject Classification: 62H30 (Primary); 65K10, 91C20, 91D30, 94C15 (Secondary)

Abstract

In network science, one of the significant and challenging subjects is the detection of communities. Modularity [1] is a measure of community structure that compares connectivity in the network with the expected connectivity in a graph sampled from a random null model. Its optimisation is a common approach to tackle the community detection problem. We present a new method for modularity maximisation, which is based on the observation that modularity can be expressed in terms of total variation on the graph and signless total variation on the null model. The resulting algorithm is of Merriman–Bence–Osher (MBO) type. Different from earlier methods of this type, the new method can easily accommodate different choices of the null model. Besides theoretical investigations of the method, we include in this paper numerical comparisons with other community detection methods, among which the MBO-type methods of Hu *et al.* [2] and Boyd *et al.* [3], and the Leiden algorithm [4].

1. Introduction

A network is a graph structure that depicts intricate systems as nodes and edges, where nodes represent objects and edges express their relationships. Edge weights can quantify the strength of these relationship, with higher edge weights indicating a stronger connection. Complex real-world systems, such as urban transportation networks, airline networks, computer communication networks, and social networks, are characterised by their members' relationships. It is a daunting, if not impossible, task to understand the structure of a network through direct (visual) observation, especially when the numbers of nodes and edges are large. It is important, therefore, to be able to accurately and efficiently detect relevant characteristics of networks.

A typical characteristic of networks is their community structure and its detection involves partitioning the set of nodes into different communities (or clusters). Intuitively a network has a community structure if the node set can be partitioned in such a way that nodes within each resulting cluster are more likely to be connected to each other than to nodes in other clusters. Different mathematically precise measures have been proposed in the literature to capture this intuitive notion. In this paper, we will focus on modularity [56, 57], which, for a given partitioning of the node set, quantifies the difference between the actual connectivity structure within each cluster and the expected connectivity structure based on a random null model. We give the precise definition in Section 2.3.

Community detection is of great theoretical and practical value for understanding the topology and predicting the behaviour of real-world networks and has been widely used in many fields, such as protein

© The Author(s), 2024. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

function prediction [60] and criminology [38]. The formulation of modularity suggests that its maximisation over all possible partitions of the node set allows one to detect communities in a given network. Modularity optimisation, however, is an NP-hard problem [9]; thus, numerous algorithms have been developed to approximately optimise modularity, including extremal optimisation [21], greedy algorithms such as the Clauset–Newman–Moore algorithm (CNM) [19], the Louvain algorithm [6], the Leiden algorithm [71], simulated annealing [28], spectral methods [26, 56], and a proximal gradient method [67].

Hu *et al.* [33] presented a total variation (TV)-based approach for optimising network modularity using a Merriman–Bence–Osher (MBO) type scheme. The MBO scheme, originally formulated as an efficient way to approximate flows by mean curvature in Merriman *et al.* [50, 51] and co-opted as a fast method for approximately solving graph classification problems by Merkurjev *et al.* [49], is an iterative algorithm that combines short-time (linear) dynamics with thresholding. By changing the specifics of the dynamics, different MBO-type schemes can be constructed and Boyd *et al.* [7] suggested an alternative MBO scheme for modularity optimisation which, unlike the scheme of [33], is based on an underlying convex approximation of modularity.

One significant advantage of MBO-type schemes is their flexibility to incorporate extra data or constraints into the scheme, for example, a fidelity-forcing term based on training data in the linear-dynamics step as in Budd *et al.* [12] or a mass-conservation constraint in the thresholding step as in Budd and van Gennip [11]. The underlying models that form the basis of MBO schemes also make rigorous analysis possible, often help interpretability of the method, and amplify the impact of even small numbers of training data. Moreover, the general form of MBO schemes – linear dynamics alternated with nonlinear thresholding steps – makes them amenable to incorporation into artificial neural networks, as in Liu *et al.* [45]. In the current paper, we focus on MBO schemes that only use the available weighted graph structure and no additional training data or mass constraints.

1.1. Contributions

Our main contribution is the development and rigorous analysis of a new MBO-type method for modularity optimisation, which we name the modularity MBO (MMBO) method. From a theoretical point of view, our method distinguishes itself from the prior MBO-type methods in [33] and [7] in that the influence of the chosen null model on the algorithm is explicit. This is due to the fact that we reformulate the modularity objective function in terms of a total variation functional on the 'observed' network whose communities we aim to detect and a signless total variation functional on the expected network under the null model. This signless total variation was a key ingredient in the maximum cut algorithm in [39]. Modularity optimisation can thus be interpreted as balancing a minimum cut problem on the observed network with a maximum cut problem on the expected network under the null model. This also allows for the easy adaptation of the new method to the use of different null models in the modularity function.[1]

We perform an in-depth theoretical study of the (various variants of the) linear operator L_{mix} that appears in the linear-dynamics step of our MMBO algorithm, guaranteeing that the algorithm is well defined. Besides allowing six variants of the linear operator, we also formulate two different variants of the MMBO algorithm that differ in the method they use to numerically compute the linear-dynamics step. Finally, we test our method on networks formed by the MNIST handwritten digits data set [43], a stochastic block model (SBM) [32] and a 'two cows' image [4]. We compare our method with the modularity optimisation methods of Clauset *et al.* [19] (CNM), Blondel *et al.* [6] (Louvain), Traag *et al.*

[1] Although some of the theoretical guarantees we give in this paper require the node degrees under the null model to be equal to those in the observed network, there are no *a priori* reasons known to the authors, why the new method should fail to work with null models that do not satisfy this condition. We tested the method with one such null model (an Erdős–Rényi null model) but chose not to include the results to curb the length of the paper.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

[71] (Leiden), Hu *et al.* [33] and Boyd *et al.* [7], as well as with spectral clustering [37], which was developed for graph clustering, but not specifically for modularity optimisation. Because of our focus on methods that do not use any training data, we do not compare with artificial-neural-network-based methods.

The contributions of this paper are as follows:

- a reformulation of modularity in terms of (signless) total variations functions (Section 3);
- the development of a new MBO-type method for modularity optimisation, which does not require a specific form of the null model (Section 5);
- a rigorous analysis of various aspects of the method, such as the matrices involved (Sections 3–5);
- the identification of different operators for the linear thresholding step following a careful consideration of the inner products underlying the method (Sections 4–5);
- comparative computational tests, which not only show the performance of the new method in relation to existing methods measured according to the obtained modularity scores and various extrinsic clustering quality scores but also serve as a replication of results for the existing methods (Section 7);
- an empirical investigation of the dependence of the methods by Hu *et al.,* by Boyd *et al.,* and the new methods on the number of eigenvectors of the linear operator that are used and the number of iterations of the algorithm (Section 7).

1.2. Paper outline

The remainder of the paper is organised as follows. In Section 2, we introduce the mathematical preliminaries that we need; in particular in Section 2.3, we (re)acquaint the reader with the modularity function.

The reformulation of modularity in terms of a total variation and signless total variation function is key to our method. We present this reformulation in Section 3, both for the case of two communities and multiple (i.e., more than two) communities.

On our way to formulating an MBO-type scheme for modularity optimisation, we require a relaxation of the original problem. This we achieve via the Ginzburg–Landau (GL) diffuse-interface techniques that we describe in Section 4.

After this, the stage is ready for the introduction of our MMBO schemes for binary community detection and multi-community detection in Section 5. The details of the numerical implementations of these schemes are discussed in Section 6, such as the Nyström extension with QR decomposition [4, 12, 23, 58] in Section 6.4, which is employed to efficiently compute the leading eigenvalues and eigenvectors of our operators.

In Section 7, we evaluate the performance of our method on synthetic and real-world networks, not only in terms of modularity optimisation and run time but also according to various performance metrics (described in Section 7.1.3) that compare the algorithms' output with ground truth community structures that are available for the data sets we use for our tests.

We close the main part of the paper in Section 8 with some conclusions and suggestions for future research.

In Appendix A, we establish some properties of two multi-well potentials. The other appendices provide theoretical results regarding the spectra of the operators we use in the linear-dynamics step of our MMBO scheme. Appendices B and C present deferred proofs for lemmas from Sections 5 and 6, respectively, while Appendix D investigates the consequences that Weyl's inequality and a rank-one matrix update theorem have for our operators.

Notation. Table 1 contains an overview of notation that is frequently used in this paper.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 1. *Summary of frequently used symbols*

Notation list			
Symbol	Description	Symbol	Description
$\mathcal{A}$	a partition of V	P	null model
c_i	the community of which node i is a member	P^{NG}	Newman–Girvan null model
$(d_C)_i$	the (weighted) degree of node i based on adjacency matrix C	$Pt(K)$	set of matrices encoding a partition of V into K subsets
D_C	degree matrix based on C	$\mathcal{Q}_\gamma$	modularity
$e^{(k)}$	row vector with $e_k^{(k)} = 1$ and, if $l \neq k$, $e_l^{(k)} = -1$	Q_C	signless Laplacian based on C
E	edge set	$Q_{C_{\mathrm{sym}}}$	signless symmetric normalised Laplacian based on C
ε	parameter of the GL functional	$Q_{C_{\mathrm{rw}}}$	signless random walk Laplacian based on C
η	parameter in stopping criterion	τ	time step in MBO scheme
f_ε	graph Ginzburg–Landau functional	TV_C	graph total variation based on C
f_ε^+	signless graph Ginzburg–Landau functional	TV_C^+	graph signless total variation based on C
G	graph	$\mathcal{TV}_C$	graph total variation for vector-valued functions based on C
γ	resolution parameter	$\mathcal{TV}_C^+$	graph signless total variation for vector-valued functions based on C
I	identity matrix	U	node-cluster association matrix
K	number of (possibly empty) clusters	U_{*l}	l^{th} column of U
Λ	eigenvalue matrix	U_{j*}	j^{th} row of U
L_{Boyd}	the linear operator in the MBO scheme of [7]	Φ	double-well potential
$L_{\mathrm{Boyd,rw}}$	random walk normalised variant of L_{Boyd}	Φ_{mul}	multi-well potential
$L_{\mathrm{Boyd,sym}}$	symmetrically normalised variant of L_{Boyd}	V	node set
L_{Hu}	the linear operator in the MBO scheme of [33]	$\mathcal{V}$	set of node functions $u : V \to \mathbb{R}$
$L_{\mathrm{Hu,rw}}$	random walk normalised variant of L_{Hu}	$\mathcal{V}_{\mathrm{bin}}$	set of node functions $u : V \to \{-1, +1\}$
$L_{\mathrm{Hu,sym}}$	symmetrically normalised variant of L_{Hu}	$\mathrm{vol}_C(S)$	the volume of $S \subset V$ based on C
L_{mix}	one of six possible linear operators in our MMBO scheme (see (38))	W	adjacency matrix of the to-be-partitioned graph
L_C	graph Laplacian based on C	X	eigenvectors matrix
$L_{C_{\mathrm{sym}}}$	symmetric normalised Laplacian based on C	$\langle \cdot, \cdot \rangle$	Euclidean inner product
$L_{C_{\mathrm{rw}}}$	random walk Laplacian based on C	$\langle \cdot, \cdot \rangle_C$	degree-weighted inner product based on C
m	the number of the eigenvalues used for the diffusion step	$\|\| \cdot \|\|_1$	Taxicab norm
$\mathbb{N}$	natural numbers excluding 0	$\|\| \cdot \|\|_2$	Euclidean norm
N_t	number of time steps in Euler scheme	$\| \cdot \|_F$	Frobenius norm
$\mathcal{N}_C(i)$	set of neighbours of node i based on C	$\| \cdot \|_\infty$	infinity-operator norm for matrices

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

2. Mathematical preliminaries

In this section, we introduce basic terminology and derive a new formulation of modularity maximisation in terms of the minimisation of a combination of graph total variation and graph signless total variation.

2.1. Graphical framework

In this paper, we consider connected, edge-weighted, undirected graphs $G = (V, E, \omega)$ with a node set $V = \{1, \ldots, |V|\}$, an edge set $E = \{(i,j)\}_{i,j=1}^{|V|}$, and edge weight ω_{ij} between node i and node j. The (weighted) adjacency matrix $W = (\omega_{ij})_{i,j=1}^{|V|}$ is a symmetric matrix whose entries ω_{ij} are zero if $i = j$ or $(i,j) \notin E$, and positive[2] otherwise. Consequently, we can consider a given adjacency matrix W (with non-negative entries and zeros on the diagonal) as defining the edge structure of the graph. Graphs with non-negative edge weights are known as unsigned graphs. We consider unweighted graphs as special cases of weighted graphs for which, for all $i, j \in V$, $\omega_{ij} \in \{0, 1\}$.

We will reserve the notation W for the adjacency matrix of the connected graph whose nodes we wish to cluster. Along the way we also encounter graphs defined by other adjacency matrices; therefore in our definitions of adjacency-matrix-dependent quantities, we will use the (dummy) symmetric matrix $C \in [0, \infty)^{|V|\times|V|}$ with entries c_{ij}. We note that we do not require the diagonal entries c_{ii} to be zero, as is sometimes required in other sources (i.e., we allow the graph defined by C to have self-loops).

We denote the set of neighbours of node $i \in V$ with respect to the adjacency matrix C by:

$$\mathcal{N}_C(i) := \{j \in V : c_{ij} > 0\}.$$

The (weighted) degree of node i with respect to the adjacency matrix C is

$$(d_C)_i := \sum_{j\in V} c_{ij}. \tag{1}$$

The diagonal degree matrix D_C of C has entries $(D_C)_{ii} = (d_C)_i$. The maximum and minimum node degrees with respect to C are

$$d_{C,\max} := \max_{i\in V} (d_C)_i \quad \text{and} \quad d_{C,\min} := \min_{i\in V} (d_C)_i,$$

respectively. The volume (i.e., total edge weight) of a subset $S \subset V$ with respect to C is defined as:

$$\mathrm{vol}_C(S) := \sum_{i\in S} (d_C)_i = \sum_{i\in S, j\in V} c_{ij}.$$

In particular, $\mathrm{vol}_C(V)$ is also called the volume of the graph with adjacency matrix C.

We define the set of real-valued node functions:

$$\mathcal{V} := \{u : V \to \mathbb{R}\}.$$

Since a function $u \in \mathcal{V}$ is fully determined by its values $u_1, \ldots, u_{|V|}$ at the (finitely many) nodes, we may equivalently interpret u as a column vector $(u_1, \ldots, u_{|V|})^T \in \mathbb{R}^{|V|}$. We will freely make use of both the interpretation as function and as vector and do not distinguish between the two in our notation. As a consequence, we can represent linear operators that act on functions $u : V \to \mathbb{R}$ by $|V|$-by-$|V|$ real matrices. Also in this context, we will not distinguish between the operator and matrix interpretations in our notation.

[2]We will use 'non-negative' to contrast with 'positive' regarding the inclusion of the number zero.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

The standard (Euclidean) inner product $\langle \cdot, \cdot \rangle$ on $\mathbb{R}^{|V|}$ is defined as $\langle u, v \rangle := u^T v$. The norms $|| \cdot ||_1$ and $|| \cdot ||_2$ are the taxicab norm (i.e., 1-norm) and the Euclidean norm (i.e., 2-norm), respectively; that is, for vectors $w \in \mathbb{R}^n$,[3]

$$\|w\|_1 := \sum_{i=1}^{n} |w_n| \quad \text{and} \quad \|w\|_2 := \left(\sum_{i=1}^{n} w_n^2 \right)^{\frac{1}{2}}.$$

If C does not contain a row with only zeros, that is, if all row sums $(d_C)_i$ are positive, we define the C-degree-weighted inner product to be

$$\langle u, v \rangle_C := \sum_{i \in V} u_i v_i (d_C)_i. \tag{2}$$

For future reference we note that, if C and $\tilde{C}$ are two adjacency matrices with positive row sums, then

$$\langle u, v \rangle_C = \langle D_{\tilde{C}}^{-1} D_C u, v \rangle_{\tilde{C}}. \tag{3}$$

Given an adjacency matrix C, and a function $u : V \to \mathbb{R}$, we define the graph total variation (TV_C) and graph signless total variation (TV_C^+) as:

$$TV_C(u) := \frac{1}{2} \sum_{i,j \in V} c_{ij} |u_i - u_j| \quad \text{and} \quad TV_C^+(u) := \frac{1}{2} \sum_{i,j \in V} c_{ij} |u_i + u_j|.$$

A vector-valued node function $u = (u^{(1)}, \ldots, u^{(K)}) : V \to \mathbb{R}^K$, where $u^{(l)}$ is the l^{th} component of u, can be interpreted as a matrix $U \in \mathbb{R}^{|V| \times K}$, with elements $U_{il} := u_i^{(l)}$. We write U_{*l} for the l^{th} column of U; thus, the column vector U_{*l} is the vector representation of the function $u^{(l)} : V \to \mathbb{R}$. We write U_{j*} for U's j^{th} row. As with the vector interpretation of real-valued node functions above, we freely use the interpretation as function or as matrix, as suits our purpose.

For vector-valued node functions u, we generalise the definition of graph total variation and graph signless total variation to

$$\mathcal{TV}_C(u) := \sum_{l=1}^{K} TV_C(u^{(l)}) = \frac{1}{2} \sum_{l=1}^{K} \sum_{i,j \in V} c_{ij} \left| u_i^{(l)} - u_j^{(l)} \right| = \frac{1}{2} \sum_{l=1}^{K} \sum_{i,j \in V} c_{ij} \left| U_{il} - U_{jl} \right|,$$

$$\mathcal{TV}_C^+(u) := \sum_{l=1}^{K} TV_C^+(u^{(l)}) = \frac{1}{2} \sum_{l=1}^{K} \sum_{i,j \in V} c_{ij} \left| u_i^{(l)} + u_j^{(l)} \right| = \frac{1}{2} \sum_{l=1}^{K} \sum_{i,j \in V} c_{ij} \left| U_{il} + U_{jl} \right|.$$

For a matrix $U \in \mathbb{R}^{n \times p}$ with entries U_{ij}, its C-Frobenius norm (with $C \in \mathbb{R}^{n \times n}$ symmetric positive definite[4]) is given by:

$$\|U\|_{\text{Fr},C} := \sqrt{\text{tr}(U^T C U)} = \sqrt{\sum_{j=1}^{p} \sum_{k,l=1}^{n} C_{kl} U_{kj} U_{lj}},$$

where tr denotes the trace of a square matrix. If C is the identity matrix I, this reduces to the standard Frobenius norm $\|U\|_{\text{Fr}} := \|U\|_{\text{Fr},I}$. Because the standard Frobenius norm is submultiplicative, that is,

[3] Or $w \in \mathbb{C}^n$, where we need it. These norms may be applied to column vectors or row vectors, as the context demands.

[4] We note that, if C is a symmetric positive definite matrix, $\| \cdot \|_{\text{Fr},C}$ indeed defines a norm, since $\|U\|_{\text{Fr},C} = \|C^{\frac{1}{2}} U\|_{\text{Fr}}$, where $C^{\frac{1}{2}}$ denotes the unique symmetric positive definite square root of C. Since the Frobenius norm $\| \cdot \|_{\text{Fr}}$ is a norm and $C^{\frac{1}{2}} U = 0$ if and only if $U = 0$, also $\| \cdot \|_{\text{Fr},C}$ is a norm.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

for all matrices $U_1 \in \mathbb{R}^{n\times p}$ and $U_2 \in \mathbb{R}^{p\times q}$, $\|U_1U_2\|_{\mathrm{Fr}} \le \|U_1\|_{\mathrm{Fr}}\|U_2\|_{\mathrm{Fr}}$, the C-Frobenius norm satisfies the following property:

$$\|U_1U_2\|_{\mathrm{Fr},C} = \|C^{\frac{1}{2}}U_1U_2\|_{\mathrm{Fr}} \le \|C^{\frac{1}{2}}U_1\|_{\mathrm{Fr}}\|U_2\|_{\mathrm{Fr}} = \|U_1\|_{\mathrm{Fr},C}\|U_2\|_{\mathrm{Fr}}. \tag{4}$$

The infinity operator norm of the matrix U is given by:

$$\|U\|_\infty := \max_{i\in\{1,\dots,n\}} \sum_{j=1}^{p} |U_{ij}|.$$

Given $K \in \mathbb{N}$, let $\mathcal{A} = \{A_l\}_{l=1}^K$ be a multiset[5] of pairwise disjoint subsets of V that partition the node set V, that is, $V = \bigcup_{l=1}^K A_l$ and $A_{l_1} \cap A_{l_2} = \emptyset$ if $l_1 \neq l_2$. Note that A_l could be empty, so the number of non-empty sets in $\mathcal{A}$ is at most K. We call two partitions (with possibly different numbers of elements) equivalent, if every element of their symmetric difference is $\emptyset$. The canonical representative of an equivalence class of partitions is the unique partition in the equivalence class that does not contain any copy of $\emptyset$. Each canonical representative $\mathcal{A}$ is in bijective correspondence to a node assignment $c : V \to \{1,\dots,|\mathcal{A}|\}$: given a canonical representative $\mathcal{A} = \{A_l\}_{l=1}^K$, define $c_j = l$ if and only if $j \in A_l$; conversely, given a node assignment $c : V \to \{1,\dots,K\}$, define $A_l = \{j \in V : c_j = l\}$ for $l \in \{1,\dots,K\}$.

We will refer to the sets A_l in a partition as the communities defined by that partition. Also the terms 'clusters' or 'classes' may be used. We use an indicator function δ with $\delta(c_i, c_j) = 1$ if nodes i and j are in the same community and $\delta(c_i, c_j) = 0$ otherwise.

2.2. Laplacians for unsigned graphs

In this section, we define graph Laplacians for unsigned graphs determined by an adjacency matrix C.

For a graph with weighted adjacency matrix C, we define the graph Laplacian matrix L_C as [16]:

$$(L_C)_{ij} := \begin{cases} (d_C)_i - c_{ii}, & \text{if } i = j, \\ -c_{ij}, & \text{otherwise.} \end{cases}$$

We include the dependence on C explicitly in the notation, since we require graph Laplacians for various different graphs.

The Laplacian matrix can be written as:

$$L_C = D_C - C.$$

If D_C is invertible, the random walk graph Laplacian matrix $L_{C_{\mathrm{rw}}}$ and the symmetrically normalised graph Laplacian matrix $L_{C_{\mathrm{sym}}}$ are given by:

$$L_{C_{\mathrm{rw}}} := D_C^{-1}L_C = I - D_C^{-1}C,$$

$$L_{C_{\mathrm{sym}}} := D_C^{-\frac{1}{2}}L_C D_C^{-\frac{1}{2}} = I - D_C^{-\frac{1}{2}}CD_C^{-\frac{1}{2}},$$

respectively. It is well known that the random walk graph Laplacian and symmetrically normalised graph Laplacian have the same eigenvalues [74].

[5] A multiset is a generalisation of the concept of set to allow for multiple copies of the same element. Formally, it can be thought of as a set of pairs (x, y), where $y \in \mathbb{N}$ is used to distinguish different copies of x, for example, $\{a, a, b, b, b\} = \{(a, 1), (a, 2), (b, 1), (b, 2), (b, 3)\}$. For a multiset M, the notation $x \in M$ means that there exists a $y \in \mathbb{N}$ such that $(x, y) \in M$. We require $\mathcal{A}$ to be a multiset and thereby deviate from the usual definition of partition, because we wish to allow $\mathcal{A}$ to contain multiple copies of the empty set (and only of the empty set, as follows from the requirement that the elements of the multiset be pairwise disjoint subsets of V).

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

In the special case that $C = W$, we recall that $G = (V, E, \omega)$ is connected, and in particular there is no isolated node. Thus, for all $i \in V$, $(d_W)_i > 0$ and hence the matrix D_W is invertible.

Let $u \in \mathbb{R}^{|V|}$. We compute, for $i \in V$,

$$(L_C u)_i = \sum_{j \in V} c_{ij} \left(u_i - u_j\right) \quad \text{and} \quad \left(L_{C_{\mathrm{rw}}} u\right)_i = \frac{1}{(d_C)_i} \sum_{j \in V} c_{ij}(u_i - u_j), \tag{5}$$

where we require $(d_C)_i > 0$ for the second computation. We note that any self-loops (i.e., $c_{ii} > 0$) do not contribute to the images of the Laplacian L_C but will contribute to the degree normalisation in $L_{C_{\mathrm{rw}}}$.

Lemma 2.1. *Let $C \in [0, \infty)^{|V| \times |V|}$ be a symmetric matrix. For parts (b) and (c) below, additionally assume that D_C is invertible.*

(a) The graph Laplacian (matrix) L_C is self-adjoint with respect to the Euclidean inner product, that is, for all $u, v \in \mathcal{V}$,

$$\langle L_C u, v \rangle = \langle u, L_C v \rangle.$$

It is also positive semidefinite with respect to the Euclidean inner product[6], that is, for all $u \in \mathcal{V}$,

$$\langle L_C u, u \rangle \geq 0.$$

(b) The symmetrically normalised graph Laplacian (matrix) $L_{C_{sym}}$ is self-adjoint and positive semidefinite with respect to the Euclidean inner product, that is, for all $u, v \in \mathcal{V}$,

$$\langle L_{C_{sym}} u, v \rangle = \langle u, L_{C_{sym}} v \rangle \quad \text{and} \quad \langle L_{C_{sym}} u, u \rangle \geq 0,$$

respectively.

(c) The random walk graph Laplacian (matrix) $L_{C_{rw}}$ is self-adjoint and positive semidefinite with respect to the C-degree-weighted inner product, that is, for all $u, v \in \mathcal{V}$,

$$\langle L_{C_{rw}} u, v \rangle_C = \langle u, L_{C_{rw}} v \rangle_C \quad \text{and} \quad \langle L_{C_{rw}} u, u \rangle_C \geq 0,$$

respectively.

Proof. Let $u, v \in \mathcal{V}$.

It follows from the symmetry of C and (5) that L_C is self-adjoint with respect to the Euclidean inner product:

$$\langle u, L_C v \rangle = \sum_{i,j \in V} c_{ij} u_i (v_i - v_j) = \sum_{i,j \in V} \left(c_{ij} u_i v_i - c_{ji} u_j v_i\right) = \sum_{i,j \in V} c_{ij} v_i (u_i - u_j) = \langle L_C u, v \rangle.$$

Interchanging the indices i and j in this calculation shows in a straightforward way that L_C is positive semidefinite:

$$\langle L_C u, u \rangle = \frac{1}{2}\left(\langle L_C u, u \rangle + \langle u, L_C u \rangle\right) = \frac{1}{2} \sum_{i,j \in V} c_{ij} (u_i - u_j)^2 \geq 0. \tag{6}$$

Similarly, the symmetrically normalised graph Laplacian $L_{C_{\mathrm{sym}}}$ is self-adjoint with respect to the Euclidean inner product, since

$$\langle u, L_{C_{\mathrm{sym}}} v \rangle = \langle D_C^{-\frac{1}{2}} u, L_C D_C^{-\frac{1}{2}} v \rangle = \langle L_C D_C^{-\frac{1}{2}} u, D_C^{-\frac{1}{2}} v \rangle = \langle L_{C_{\mathrm{sym}}} u, v \rangle,$$

[6] Perhaps a more correct way to express this would be to say that the bilinear form $(u, v) \mapsto \langle L_C u, v \rangle$ is positive semidefinite.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

and it is positive semidefinite with respect to the same inner product:

$$\langle L_{C_{\text{sym}}} u, u\rangle = \frac{1}{2}\big(\langle L_{W_{\text{sym}}} u, u\rangle + \langle u, L_{W_{\text{sym}}} u\rangle\big) = \frac{1}{2}\sum_{i,j\in V} c_{ij}\left(\frac{u_i}{\sqrt{(d_C)_i}} - \frac{u_j}{\sqrt{(d_C)_j}}\right)^2 \geq 0. \tag{7}$$

We recall that, for all $i \in V$, $(d_C)_i > 0$, because the diagonal matrix D_C is invertible.

Finally, using (5) we compute

$$\langle u, L_{C_{\text{rw}}} v\rangle_C = \langle u, D_C L_{C_{\text{rw}}} v\rangle = \langle u, L_C v\rangle = \langle L_C u, v\rangle = \langle D_C L_{W_{\text{rw}}} u, v\rangle = \langle L_{C_{\text{rw}}} u, v\rangle_C$$

and

$$\langle L_{C_{\text{rw}}} u, u\rangle_C = \frac{1}{2}\sum_{i,j\in V} c_{ij}\left(u_i - u_j\right)^2 \geq 0. \tag{8}$$

□

The signless graph Laplacian (matrix) Q_C for a graph with weighted adjacency matrix C, and its random walk and symmetrically normalised variants $Q_{C_{\text{rw}}}$ and $Q_{C_{\text{sym}}}$, respectively, are defined as:

$$\begin{aligned} Q_C &:= D_C + C, \\ Q_{C_{\text{rw}}} &:= D_C^{-1} Q_C = I + D_C^{-1} C, \\ Q_{C_{\text{sym}}} &:= D_C^{-\frac{1}{2}} Q_C D_C^{-\frac{1}{2}} = I + D_C^{-\frac{1}{2}} C D_C^{-\frac{1}{2}}. \end{aligned}$$

Lemma 2.2. *Let $C \in [0,\infty)^{|V|\times|V|}$ be a symmetric matrix. For parts (b) and (c) below, additionally assume that D_C is invertible.*

(a) The signless graph Laplacian (matrix) Q_C is self-adjoint and positive semidefinite with respect to the Euclidean inner product, that is, for all $u, v \in \mathcal{V}$,

$$\langle Q_C u, v\rangle = \langle u, Q_C v\rangle \quad \text{and} \quad \langle Q_C u, u\rangle \geq 0,$$

respectively.

(b) The symmetrically normalised signless graph Laplacian (matrix) $Q_{C_{sym}}$ is self-adjoint and positive semidefinite with respect to the Euclidean inner product, that is, for all $u, v \in \mathcal{V}$,

$$\langle Q_{C_{sym}} u, v\rangle = \langle u, Q_{C_{sym}} v\rangle \quad \text{and} \quad \langle Q_{C_{sym}} u, u\rangle \geq 0,$$

respectively.

(c) The random walk signless graph Laplacian (matrix) $Q_{C_{rw}}$ is self-adjoint and positive semidefinite with respect to the C-degree-weighted inner product, that is, for all $u, v \in \mathcal{V}$,

$$\langle Q_{C_{rw}} u, v\rangle_C = \langle u, Q_{C_{rw}} v\rangle_C \quad \text{and} \quad \langle Q_{C_{rw}} u, u\rangle_C \geq 0,$$

respectively.

Proof. The proofs are analogous to the proofs of the statements in Lemma 2.1. For future reference, we do note that, for all $u \in \mathcal{V}$,

$$(Q_C u)_i = \sum_{j\in V} c_{ij}\left(u_i + u_j\right) \quad \text{and} \quad \left(Q_{C_{\text{rw}}} u\right)_i = \frac{1}{(d_C)_i}\sum_{j\in V} c_{ij}(u_i + u_j), \tag{9}$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

and

$$\langle Q_C u, u\rangle = \frac{1}{2}\sum_{i,j\in V} c_{ij}(u_i + u_j)^2 \geq 0, \tag{10}$$

$$\langle Q_{C_{\text{sym}}} u, u\rangle = \frac{1}{2}\sum_{i,j\in V} c_{ij}\left(\frac{u_i}{\sqrt{(d_C)_i}} + \frac{u_j}{\sqrt{(d_C)_j}}\right)^2 \geq 0, \tag{11}$$

$$\langle Q_{C_{\text{rw}}} u, u\rangle_C = \frac{1}{2}\sum_{i,j\in V} c_{ij}\left(u_i + u_j\right)^2 \geq 0. \tag{12}$$

□

Remark 2.3. Differently from what we observed for the graph Laplacian L_C in (5), the presence of any self-loops in the graph with adjacency matrix C will influence the images of both signless graph Laplacians in (9).

2.3. Review of the modularity function

'Community structure' is not a single well-defined concept. It attempts to capture the notion of partitioning a collection of individuals[7] (set of nodes) into meaningful clusters (classes and communities). What is meaningful depends on the context. Attempts to quantify 'community structure' tend to fall into one of two categories: comparisons with an externally available reference clustering, which for easy reference we will call the 'ground truth' partition,[8] and applications of mathematically defined 'measures'[9] of community structure that do not require a ground truth. The approaches and measures from these categories are called extrinsic and intrinsic, respectively.

Extrinsic comparisons are used mostly when testing new methods on benchmark data sets for which the 'preferred' community structure is known through other means, such as the MNIST data set of handwritten digits (see Section 7.2). It may also be useful when investigating if the features that determine the network structure can be used to detect a community structure which is defined in terms of other features, for example, when a collaboration network of scientists is used in an attempt to construct a clustering that agrees with the areas of expertise of the scientists.

Intrinsic approaches have the advantages that no known ground truth is needed and that the mathematical formulation in terms of an optimisation problem for a well-defined measure of community structure can help in algorithm development. Once the mathematical problem has been formulated, the problem also becomes independent of context, although it still depends on the context how useful (in a practical, real-world sense) a given mathematical formulation is.

This work mainly falls in the second category, as we will use modularity [57, 56] as a measure of community structure and develop a new algorithm to approach the modularity optimisation problem. We will, however, also dip our toes into the first category, when we judge the outcomes of our algorithm not only by their modularity scores, but also through comparisons with ground truth community structures where these are available.

In the most general form that we encounter in this work, the definition of the *modularity* of a partition $\mathcal{A}$ of the node set of a graph $G = (V, E, \omega)$ is

$$\mathcal{Q}(\mathcal{A}; W, P) := \frac{1}{\text{vol}_W(V)}\sum_{i,j\in V}\left(\omega_{ij} - p_{ij}\right)\delta(c_i, c_j). \tag{13}$$

[7] Although the terminology 'communities' and 'individuals' suggests a context in which the individuals are people, there is no reason to restrict ourselves to such settings *a priori*.

[8] In some contexts, the term 'ground truth' appears to promise more than is justified. In this paper, we intend it to be equivalent to the phrase 'reference clustering,' even if some of its other connotations are not entirely out of place for our classification, SBM, and image segmentation examples in Section 7.

[9] Not in the measure-theoretic sense.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

The matrix $P = (p_{ij})_{i,j=1}^{|V|}$ encodes the expected edge weight p_{ij} between nodes i and j under a given *null model*, that is, a random graph satisfying some constraints. Since we are interested in undirected graphs with non-negative edge weights in this work, we assume P to be symmetric and to have non-negative entries. We note that there is no restriction for the diagonal elements of P to be zero. The null model can be thought of as describing graph structures that one would expect to see if there were *no* community structure present.

The modularity optimisation problem consists of finding $\operatorname{argmax}_{\mathcal{A}} \mathcal{Q}(\mathcal{A}; W, P)$. For simplicity of notation, where we write $\operatorname{argmax}_{\mathcal{A}}$ or $\max_{\mathcal{A}}$, it is implicitly assumed that the maximum is taken over all partitions $\mathcal{A}$ of V. We emphasise that K, that is, the number of sets in $\mathcal{A}$, is not fixed: finding an optimal K is part of the optimisation problem. Because we do not assume that the sets in $\mathcal{A}$ are non-empty, solutions of the optimisation problem are necessarily non-unique, as $\mathcal{Q}(\mathcal{A}; W, P) = \mathcal{Q}(\mathcal{A} \cup \{\emptyset\}; W, P)$.[10]

One of the commonly used null models is the Newman–Girvan (NG) model [57], which is a random *unweighted* graph in which edges are assigned unbiasedly at random to pairs of distinct nodes, under the constraint that the resulting node degrees are equal to the degrees $(d_W)_i$ in the graph G. The resulting expected edge weights are (approximately[11]) $p_{ij}^{NG} = \frac{(d_W)_i (d_W)_j}{\mathrm{vol}_W(V)}$, which are used as entries for the matrix P^{NG}. Modularity with the NG null model thus can be written as:

$$\mathcal{Q}(\mathcal{A}; W, P^{\mathrm{NG}}) = \frac{1}{\mathrm{vol}_W(V)} \sum_{i,j \in V} \left(\omega_{ij} - \frac{(d_W)_i (d_W)_j}{\mathrm{vol}_W(V)} \right) \delta(c_i, c_j). \tag{14}$$

We note indeed that the degrees under the null model are the same as those in G: $(d_{P^{\mathrm{NG}}})_i = \sum_{j \in V} P^{\mathrm{NG}}_{ij} = (d_W)_i$. This has some very useful consequences, for example, $\mathrm{vol}_{P^{\mathrm{NG}}}(V) = \sum_{i \in V} (d_{P^{\mathrm{NG}}})_i = \sum_{i \in V} (d_W)_i = \mathrm{vol}_W(V)$ and both $L_{P^{\mathrm{NG}}_{\mathrm{sym}}}$ and $Q_{P^{\mathrm{NG}}_{\mathrm{sym}}}$ are self-adjoint with respect to the inner product from (2) with $C = W$.

With this choice of null model $\mathcal{Q}(\mathcal{A}; W, P^{\mathrm{NG}}) = 0$ if $K = 1$, that is, if the partition is $\mathcal{A} = \{V\}$.

According to Fortunato and Barthelemy [22], there is a drawback in optimising equation (14) to find community partitions: it is difficult to find a community partition in networks that contain many small communities. It is argued that (in an unweighted graph) the number of communities K that produces the maximum modularity score is (approximately) equal to $\sqrt{\frac{\mathrm{vol}_W(V)}{2}}$. A partition with many small communities tends to have more communities than this optimal value.

To solve the above problem, Arenas [2] proposed a generalised modularity based on the Reichardt and Bornholdt method [62]:

$$\mathcal{Q}_\gamma(\mathcal{A}; W, P) := \frac{1}{\mathrm{vol}_W(V)} \sum_{i,j \in V} \left(\omega_{ij} - \gamma p_{ij} \right) \delta(c_i, c_j), \tag{15}$$

$$\mathcal{Q}_\gamma(\mathcal{A}; W, P^{\mathrm{NG}}) = \frac{1}{\mathrm{vol}_W(V)} \sum_{i,j \in V} \left(\omega_{ij} - \gamma \frac{(d_W)_i (d_W)_j}{\mathrm{vol}_W(V)} \right) \delta(c_i, c_j), \tag{16}$$

where $\gamma > 0$ is a resolution parameter [62]. The distinction between (15) and (13) is the parameter[12] γ, which allows (15) to be more flexible and find more network community partitions; we note that $\mathcal{Q}_1 = \mathcal{Q}$. Nevertheless, there are still some issues with modularity optimisation even in this case.

[10] A more interesting question is whether (or under which conditions) the maximiser is unique among canonical representatives, up to permutations of the labels l of the subsets A_l. The authors are not aware of any attempts in the literature to address this question.

[11] In fact, in the *configuration model*, which underlies the random graph described here, the expected number of edges between nodes i and j is $\frac{(d_W)_i (d_W)_j}{\mathrm{vol}_W(V) - 1}$ [55, Section 13.2]. In the NG model, however, $\mathrm{vol}_W(V)$ is used in the denominator, rather than $\mathrm{vol}_W(V) - 1$. Mathematically, this has the pleasant consequence that the degree of each node in the graph defined by adjacency matrix W is the same as its degree in the graph defined by adjacency matrix P^{NG}, as we show below. For graphs with a large volume $\mathrm{vol}_W(V)$, the difference between the value used in the NG null model and the expectation that follows from the configuration model will be small.

[12] We view γ as a separate parameter, rather than absorbing it into the matrix P, so that we can keep the interpretation of P as a matrix of expected values. Moreover, this way the useful property $D_{P^{\mathrm{NG}}} = D_W$ remains valid.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Lancichinetti and Fortunato [41] make the case that for large enough γ, the partition with maximum modularity score will split random subgraphs, which is unwanted behaviour, since random subgraphs should not be identified as having a community structure. On the other hand, they argue that for small enough γ, there will be communities that contain multiple subgraphs, even when the number of internal edges in these subgraphs is large and there is only one inter-subgraph edge. Again, this is unwanted behaviour. Moreover, they show that it is difficult and in some cases even impossible to select a value for γ that eliminates both these biases. In brief, for smaller values of γ, we expect fewer clusters than for larger values.

A strategy for sampling the range of possible resolutions is presented in Jeub *et al.* [36]. Another approach to deal with this shortcoming is to investigate the stability of communities over multiple resolution scales, as in Mucha *et al.* [54]. Since here we are primarily interested in the ability of our new algorithms to optimise modularity, rather than the appropriateness of the chosen resolution scale, we will not pursue those approaches here, and the problem of how to determine a good value for γ in any given context is a matter outside the scope of this paper.

3. Reformulation of modularity optimisation

The new method we propose in this paper is based on the observation that the modularity function can be reformulated in terms of (signless) total variation functions.

3.1. Reformulation of modularity optimisation for binary segmentation

In this subsection, we derive a new expression for the modularity $\mathcal{Q}(\mathcal{A})$ restricted to partitions $\mathcal{A} = \{A_l\}_{l=1}^{K}$ with $K = 2$, transforming the maximisation problem into a minimisation problem.

Let u be a real-valued function on the node set V, with value u_i on node i. We define the set of $\{-1, +1\}$-valued node functions as:

$$\mathcal{V}_{\text{bin}} := \big\{u : V \to \{-1, +1\}\big\}.$$

Specially, if $u \in \mathcal{V}_{\text{bin}}$, we define the sets

$$V_1 = \{i \in V : u_i = 1\} \quad \text{and} \quad V_{-1} = \{i \in V : u_i = -1\}.$$

We consider a partition $\mathcal{A} = \{A_1, A_2\}$, with $A_1 = V_1$, $A_2 = V_{-1}$, and corresponding node assignment c.

If $u_i \in \mathcal{V}_{\text{bin}}$, that is, $u_i \in \{-1, +1\}$, then $u_i^2 + u_j^2 = 2$, which implies that

$$(u_i u_j + 1) = -\frac{1}{2}(u_i - u_j)^2 + 2 \quad \text{and} \quad -(u_i u_j + 1) = -\frac{1}{2}(u_i + u_j)^2. \tag{17}$$

If $i, j \in V_1$ or $i, j \in V_{-1}$, then $\delta(c_i, c_j) = 1 = \frac{1}{2}(u_i u_j + 1)$. Similarly, if $u_i \neq u_j$, then $\delta(c_i, c_j) = 0 = \frac{1}{2}(u_i u_j + 1)$. Hence, the modularity $\mathcal{Q}$ for $K = 2$ clusters can be rewritten as:[13]

$$\begin{aligned}
\mathcal{Q}_\gamma(\mathcal{A}; W, P) &= \frac{1}{2\text{vol}_W(V)} \sum_{i,j \in V} \big(\omega_{ij} - \gamma p_{ij}\big)(u_i u_j + 1) \\
&= \frac{1}{2\text{vol}_W(V)} \sum_{i,j \in V} \omega_{ij}(u_i u_j + 1) - \frac{\gamma}{2\text{vol}_W(V)} \sum_{i,j \in V} p_{ij}(u_i u_j + 1) \\
&= -\frac{1}{4\text{vol}_W(V)} \sum_{i,j \in V} \omega_{ij}(u_i - u_j)^2 + \frac{1}{\text{vol}_W(V)} \sum_{i,j \in V} \omega_{ij} - \frac{\gamma}{4\text{vol}_W(V)} \sum_{i,j \in V} p_{ij}(u_i + u_j)^2 \\
&= -\frac{1}{\text{vol}_W(V)} \left[\frac{1}{4} \sum_{i,j \in V} \omega_{ij}(u_i - u_j)^2 + \frac{\gamma}{4} \sum_{i,j \in V} p_{ij}(u_i + u_j)^2\right] + \frac{1}{\text{vol}_W(V)} \sum_{i,j \in V} \omega_{ij}.
\end{aligned} \tag{18}$$

[13] As an interesting aside, we observe that we do not require any symmetry properties of W or P for this computation.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

For all $u \in \mathcal{V}_{\text{bin}}$, one obtains

$$(u_i - u_j)^2 = \begin{cases} 0, & \text{if } u_i = u_j, \\ 4, & \text{if } u_i \neq u_j, \end{cases} \quad \text{and} \quad 2|u_i - u_j| = \begin{cases} 0, & \text{if } u_i = u_j, \\ 4, & \text{if } u_i \neq u_j. \end{cases}$$

Therefore, if $u \in \mathcal{V}_{\text{bin}}$, then $(u_i - u_j)^2 = 2|u_i - u_j|$. Similarly, $(u_i + u_j)^2 = 2|u_i + u_j|$, if $u \in \mathcal{V}_{\text{bin}}$.

Since $\text{vol}_W(V) = \sum_{i,j \in V} \omega_{ij}$, the third term in (18) equals one; in particular, it does not depend on u. Thus,

$$\mathcal{Q}_\gamma(\mathcal{A}; W, P) = -\frac{1}{\text{vol}_W(V)} \mathcal{Q}_{\text{bin},\gamma}(u; W, P) + 1,$$

where

$$\begin{aligned} \mathcal{Q}_{\text{bin},\gamma}(u; W, P) &:= \frac{1}{4} \sum_{i,j \in V} \omega_{ij}(u_i - u_j)^2 + \frac{\gamma}{4} \sum_{i,j \in V} p_{ij}(u_i + u_j)^2 \\ &= \frac{1}{2} \sum_{i,j \in V} \omega_{ij}|u_i - u_j| + \frac{\gamma}{2} \sum_{i,j \in V} p_{ij}|u_i + u_j| \\ &= TV_W(u) + \gamma TV_P^+(u). \end{aligned} \tag{19}$$

The maximisation of modularity $\mathcal{Q}_\gamma(\mathcal{A}; W, P)$ from (15) over partitions $\mathcal{A}$ with $K = 2$ is equivalent to the minimisation of $\mathcal{Q}_{\text{bin},\gamma}(u; W, P)$ (19) over all functions $u \in \mathcal{V}_{\text{bin}}$ (with the correspondence between $\mathcal{A}$ and u introduced above).

Similarly to Newman [56], we define the modularity matrix as:

$$B_\gamma := W - \gamma P$$

and denote its entries by b_{ij}. Since W and P are assumed to be symmetric, so is B_γ. We note that B_γ does not need to have zeros on its diagonal.

If the condition $D_W = D_P$ is satisfied, as is the case, we recall from Section 2.3, if $P = P^{\text{NG}}$ is given by the NG null model, then

$$D_{B_\gamma} = (1 - \gamma) D_W. \tag{20}$$

In our considerations above, we have split the matrix B_γ into the matrix W with non-negative entries and the matrix $-\gamma P$ with non-positive entries, but we can also write $B_\gamma = B_\gamma^+ - B_\gamma^-$, where B_γ^+ and B_γ^- are $|V|$-by-$|V|$ matrices with entries

$$(b_\gamma^+)_{ij} := \max\{(b_\gamma)_{ij}, 0\} \quad \text{and} \quad (b_\gamma^-)_{ij} := -\min\{(b_\gamma)_{ij}, 0\},$$

respectively, with $(b_\gamma)_{ij} := \omega_{ij} - \gamma p_{ij}$ the entries of B_γ. Per definition B_γ^+ has non-negative entries and $-B_\gamma^-$ non-positive entries, yet in general $B_\gamma^+ \neq W$ and $B_\gamma^- \neq \gamma P$. Thus, this split will give another way to rewrite $\mathcal{Q}_\gamma$ analogously to (18):

$$\begin{aligned} \mathcal{Q}_\gamma(\mathcal{A}; W, P) &= -\frac{1}{2\text{vol}_W(V)} \left[\frac{1}{2} \sum_{i,j \in V} (b_\gamma^+)_{ij}(u_i - u_j)^2 + \frac{1}{2} \sum_{i,j \in V} (b_\gamma^-)_{ij}(u_i + u_j)^2 \right] + \frac{1}{\text{vol}_W(V)} \sum_{i,j \in V} (b_\gamma^+)_{ij} \\ &= -\frac{1}{\text{vol}_W(V)} \mathcal{Q}_{\text{bin},1}(u; B_\gamma^+, B_\gamma^-) + \frac{1}{\text{vol}_W(V)} \sum_{i,j \in V} (b_\gamma^+)_{ij}. \end{aligned}$$

Because the final term on the right-hand side does not depend on u, we see that maximising $\mathcal{Q}_\gamma(\mathcal{A}; W, P)$ over all bipartitions $\mathcal{A}$ is equivalent to minimising $\mathcal{Q}_{\text{bin},1}(u; B_\gamma^+, B_\gamma^-)$ over all $u \in \mathcal{V}_{\text{bin}}$.

In the remainder of this paper, we would like to be able to consider graph Laplacians based on B_γ^+ and B_γ^-. To be able to define random walk and symmetrically normalised (signless) graph Laplacians based on these matrices, we require the degree matrices $D_{B_\gamma^+}$ and $D_{B_\gamma^-}$ to be invertible. Additionally, for the symmetrically normalised (signless) graph Laplacians, we also require these matrices to be positive

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

semidefinite[14] so that their square roots are uniquely defined. This property is easily checked to hold, since both matrices are diagonal with non-negative entries.

The following lemma and corollary collect some useful results about their invertibility for easy reference.

Lemma 3.1. *Let $i \in V$ and $\gamma \in (0, \infty)$. Then $(d_{B_\gamma^+})_i \neq 0$ if and only if there exists a $j \in \mathcal{N}_W(i)$ such that*

$$\gamma < p_{ij}^{-1}\omega_{ij}, \tag{21}$$

where we define $p_{ij}^{-1}\omega_{ij} := +\infty$ if $p_{ij} = 0$.

Similarly, $(d_{B_\gamma^-})_i \neq 0$ if and only if there exists a $j \in \mathcal{N}_P(i)$ such that

$$\gamma \geq p_{ij}^{-1}\omega_{ij}. \tag{22}$$

Consequently, $D_{B_\gamma^+}$ is invertible if and only if

$$\gamma < \min_{i \in V} \max_{j \in \mathcal{N}_W(i)} p_{ij}^{-1}\omega_{ij}$$

and $D_{B_\gamma^-}$ is invertible if and only if

$$\gamma > \max_{i \in V} \min_{j \in \mathcal{N}_P(i)} p_{ij}^{-1}\omega_{ij}.$$

Proof. First, we assume that $(d_{B_\gamma^+})_i = \sum_{k \in V} (b_\gamma^+)_{ik} = 0$. Since all terms in the sum are non-negative, this means that, for all $k \in V$, $(b_\gamma^+)_{ik} = \max\{\omega_{ik} - \gamma p_{ik}, 0\} = 0$. In particular, this holds for all $k \in \mathcal{N}_W(i)$. Hence, for all $k \in \mathcal{N}_W(i)$, $\omega_{ik} \leq \gamma p_{ik}$. Since, $\gamma > 0$ and, per definition, for all $k \in \mathcal{N}_W(i)$, $\omega_{ik} > 0$, for all such k we get $p_{ik} > 0$ and thus $\gamma \geq p_{ik}^{-1}\omega_{ik}$. This proves the contrapositive of the first 'if' statement from the lemma.

To prove the contrapositive of the corresponding 'only if' statement, assume that, for all $k \in \mathcal{N}_W(i)$, $\gamma \geq p_{ik}^{-1}\omega_{ik}$. Since $\gamma < +\infty$ and, for all $k \in \mathcal{N}_W(i)$, $\omega_{ik} > 0$, this implies that, for such k, $p_{ik} \neq 0$. Hence, for all $k \in \mathcal{N}_W(i)$, $\gamma p_{ik} \geq \omega_{ik}$. If $k \in V \setminus \mathcal{N}_W(i)$, then $\omega_{ik} = 0$ and thus trivially $\gamma p_{ik} \geq \omega_{ik}$. This proves the statement.

The proofs of the analogous statements for $(d_{B_\gamma^-})_i$ are very similar. For the 'if' statement, we note that $(d_{B_\gamma^-})_i = 0$ implies that, for all $k \in V$, and thus in particular for all $k \in \mathcal{N}_P(i)$, $\omega_{ik} \geq \gamma p_{ik}$. Since, for all $k \in \mathcal{N}_P(i)$, $p_{ik} \neq 0$, we obtain $\gamma \leq p_{ik}^{-1}\omega_{ik}$ as required.

For the 'only if' statement, we assume that, for all $k \in \mathcal{N}_P(i)$, $\gamma \leq p_{ik}^{-1}\omega_{ik}$. Thus, for all $k \in \mathcal{N}_P(i)$, $\gamma p_{ik} \leq \omega_{ik}$ and hence $(b_\gamma^-)_{ik} = 0$. For all $k \in V \setminus \mathcal{N}_P(i)$, we have $p_{ik} = 0$ and thus $(b_\gamma^-)_{ik} = -\min\{\omega_{ij}, 0\} = 0$. Hence, $(d_{B_\gamma^-})_i = 0$.

Because $D_{B_\gamma^+}$ and $D_{B_\gamma^-}$ are real diagonal matrices, they are invertible if and only if all their diagonal elements are non-zero. By the first part of this lemma, we know this is true for $D_{B_\gamma^+}$ if and only if, for all $i \in V$ there exists a $j \in \mathcal{N}_W(i)$ for which (21) holds. Thus, it is sufficient if it holds for a $j^* \in \operatorname{argmax}_{j \in \mathcal{N}_W(i)} p_{ij}^{-1}\omega_{ij}$. To ensure the condition holds for all $i \in V$, the minimum is taken over all such i.

Similarly, by the second part of this lemma, $D_{B_\gamma^-}$ is invertible if, for all $i \in V$ there exists a $j \in \mathcal{N}_P(i)$ for which (22) holds. By a similar argument as for $D_{B_\gamma^+}$, we obtain the result. □

Corollary 3.2.

1. *Let the null model be such that $D_W = D_P$ and assume that, for all $i \in V$, there exists a $j \in V$ for which $\omega_{ij} \neq p_{ij}$. Then there exists an open interval I containing 1, such that, for all $\gamma \in I$, $D_{B_\gamma^+}$ and $D_{B_\gamma^-}$ are invertible.*
2. *In particular, if the stated assumptions hold and $\gamma = 1$, then $D_{B_\gamma^+} = D_{B_\gamma^-}$ is invertible.*

Proof. Let $i \in V$. By assumption, there exists a $j \in V$ such that $\omega_{ij} \neq p_{ij}$.

[14] In fact, if these diagonal matrices are positive semidefinite and invertible, they are in fact positive definite.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

First, we assume that $\omega_{ij} > p_{ij}$. If $j \in V \setminus \mathcal{N}_W(i)$, then $\omega_{ij} = 0$, which contradicts $p_{ij} \geq 0$. Hence, $j \in \mathcal{N}_W(i)$ and thus the right-hand side of condition (21) is strictly greater than 1.

Moreover, since by assumption $0 = (d_W)_i - (d_P)_i = \omega_{ij} - p_{ij} + \sum_{k \in V \setminus \{j\}} (\omega_{ik} - p_{ik})$, there must exist a $k \in V \setminus \{j\}$ such that $\omega_{ik} < p_{ik}$. If $k \in V \setminus \mathcal{N}_P(i)$, then $p_{ik} = 0$, which contradicts $\omega_{ik} \geq 0$, hence $p_{ik} > 0$. Thus, the right-hand side of condition (22) is strictly smaller than 1 (with $j = k$).

This proves the claim if $\omega_{ij} > p_{ij}$. If instead $\omega_{ij} < p_{ij}$, we repeat the proof above with the roles of j and k interchanged.

The final statement of the corollary follows immediately from (20) combined with the first part of this corollary for $\gamma = 1 \in I$. □

Remark 3.3. Per our discussion in Section 2.3, we note that the assumption $D_W = D_P$ is satisfied if we use the NG null model. Rather than checking the additional requirements of Corollary 3.2 part 1 explicitly, in our numerical tests in Section 7 we ensure invertibility of $D_{B_\gamma^+}$ and $D_{B_\gamma^-}$ directly by checking diagonals for zero entries.

3.2. Generalisation to multiple clusters

Next, we extend the approach of Section 3.1 to identify appropriate partitions of the node set into multiple clusters. In this subsection, we consider K to be fixed (but not necessarily equal to two). Then a partition $\mathcal{A} = \{A_l\}_{l=1}^K$ with K parts is completely described by a function $u = (u^{(1)}, \ldots, u^{(K)}) : V \to \mathbb{R}^K$, where $u_i^{(l)} = 1$ if and only if $i \in A_l$ and $u_i^{(l)} = -1$ otherwise. We can also encode this information in a matrix $U \in \mathbb{R}^{|V| \times K}$, with elements $U_{il} := u_i^{(l)}$. We denote the set of all matrices corresponding to a partition in K parts by:

$$Pt(K) := \left\{ U \in \mathbb{R}^{|V| \times K} : \forall i \in V \ \ \forall l \in \{1, \ldots, K\} \ \ U_{il} \in \{-1, 1\} \text{ and } \sum_{k=1}^{K} U_{ik} = 2 - K \right\}. \tag{23}$$

We may call such matrices partition matrices. The last condition in the definition of $Pt(K)$ guarantees that each node belongs to exactly one cluster. We recall that $\mathcal{A}$ may contain empty sets, thus, for all $l \in \{1, \ldots, K\}$, $\sum_{j \in V} u_j^{(l)} \in [-|V|, |V|] \cap \mathbb{Z}$.

Because we will require the use of (signless) total variation with respect to various different matrices, for the moment we consider an arbitrary real-valued matrix $C \in \mathbb{R}^{|V| \times |V|}$ with entries c_{ij}.

We briefly compare the current set-up with that from Section 3.1 in the case $K = 2$ with partition $\mathcal{A} = \{A_1, A_2\}$. In the notation of the current section, this partition is encoded by a matrix $U \in Pt(2)$; in the set-up of Section 3.1 we encode the same partition by a function $v \in \mathcal{V}_{\text{bin}}$. These two encodings are related via $U_{*1} = v$ and $U_{*2} = -v$ and thus

$$\begin{aligned} \mathcal{TV}_C(U) &= \frac{1}{2} \left(\sum_{i,j \in V} c_{ij} |U_{i1} - U_{j1}| + \sum_{i,j \in V} c_{ij} |U_{i2} - U_{j2}| \right) \\ &= \frac{1}{2} \left(\sum_{i,j \in V} c_{ij} |v_i - v_j| + \sum_{i,j \in V} c_{ij} |-v_i + v_j| \right) = 2TV_C(v) \end{aligned} \tag{24}$$

and similarly $\mathcal{TV}_C^+(U) = 2TV_C^+(v)$.

We return to the case of general K. For notational convenience, in the following computations we write $\tilde{U}_{ijl} := c_{ij} |U_{il} - U_{jl}|$ and $\hat{U}_{ijl} := c_{ij} |U_{il} + U_{jl}|$. For a matrix $U \in Pt(K)$, we obtain that

$$|U_{il} - U_{jl}| = \begin{cases} 0 & \text{if } i, j \in A_l \text{ or } i, j \in A_l^c, \\ 2 & \text{if } i \in A_l, j \in A_l^c \text{ or } i \in A_l^c, j \in A_l, \end{cases} \tag{25}$$

$$|U_{il} + U_{jl}| = \begin{cases} 0 & \text{if } i \in A_l, j \in A_l^c \text{ or } i \in A_l^c, j \in A_l, \\ 2 & \text{if } i, j \in A_l \text{ or } i, j \in A_l^c. \end{cases} \tag{26}$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

(We write A_l^c for the complement $V \setminus A_l^c$.) Thus, the generalised graph total variation and graph signless total variation of $U \in Pt(K)$ on a graph with adjacency matrix C can be represented as:

$$\begin{aligned}\mathcal{TV}_C(U) &= \frac{1}{2}\sum_{l=1}^{K}\left(\sum_{i,j\in A_l}\tilde{U}_{ijl} + \sum_{i\in A_l, j\in A_l^c}\tilde{U}_{ijl} + \sum_{i\in A_l^c, j\in A_l}\tilde{U}_{ijl} + \sum_{i,j\in A_l^c}\tilde{U}_{ijl}\right)\\ &= \sum_{l=1}^{K}\left(\sum_{i\in A_l, j\in A_l^c} c_{ij} + \sum_{i\in A_l^c, j\in A_l} c_{ij}\right)\end{aligned}$$

and

$$\begin{aligned}\mathcal{TV}_C^+(U) &= \frac{1}{2}\sum_{l=1}^{K}\left(\sum_{i,j\in A_l}\hat{U}_{ijl} + \sum_{i\in A_l, j\in A_l^c}\hat{U}_{ijl} + \sum_{i\in A_l^c, j\in A_l}\hat{U}_{ijl} + \sum_{i,j\in A_l^c}\hat{U}_{ijl}\right)\\ &= \frac{1}{2}\sum_{l=1}^{K}\left(\sum_{i,j\in A_l}\hat{U}_{ijl} + \sum_{i,j\in A_l^c}\hat{U}_{ijl}\right)\\ &= \sum_{l=1}^{K}\sum_{i,j\in A_l} c_{ij} + \sum_{l=1}^{K}\left(\sum_{i,j\in V} c_{ij} + \sum_{i,j\in A_l} c_{ij} - \sum_{i\in V, j\in A_l} c_{ij} - \sum_{i\in A_l, j\in V} c_{ij}\right)\\ &= \sum_{l=1}^{K}\sum_{i,j\in A_l} c_{ij} + \left(\sum_{l=1}^{K}\sum_{i,j\in A_l} c_{ij} + (K-2)\mathrm{vol}_C(V)\right)\\ &= 2\sum_{l=1}^{K}\sum_{i,j\in A_l} c_{ij} + (K-2)\mathrm{vol}_C(V).\end{aligned}$$

Using these expressions for the generalised total variation and generalised signless total variation, if $\mathcal{A}$ contains K subsets, the modularity from (13) can be written as:[15]

$$\begin{aligned}\mathcal{Q}_\gamma(\mathcal{A}; W, P) &= \frac{1}{\mathrm{vol}_W(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l}\omega_{ij} - \frac{\gamma}{\mathrm{vol}_W(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l} p_{ij}\\ &= \frac{1}{\mathrm{vol}_W(V)}\sum_{l=1}^{K}\left(\sum_{i\in A_l, j\in V}\omega_{ij} - \sum_{i\in A_l, j\in A_l^c}\omega_{ij}\right) - \frac{\gamma}{\mathrm{vol}_W(V)}\sum_{l=1}^{K}\sum_{i,j\in A_l} p_{ij}\end{aligned}$$

[15] We note that neither in this computation, nor in the previous computations for $\mathcal{TV}_C$ and $\mathcal{TV}_C^+$ in this subsection, do we require symmetry properties of C, W or P. In particular, for the fourth equality of the following computation, we use that even without symmetry we have

$$\sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^c}\omega_{ij} = \sum_{l=1}^{K}\sum_{i\in A_l^c, j\in A_l}\omega_{ij}.$$

If we do use the symmetry of W, we can further reduce $\mathcal{TV}_W(U) = 2\sum_{l=1}^{K}\sum_{i\in A_l, j\in A_l^c}\omega_{ij}$.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

$$= \frac{1}{\mathrm{vol}_W(V)} \sum_{i,j=1}^{|V|} \omega_{ij} - \frac{1}{\mathrm{vol}_W(V)} \sum_{l=1}^{K} \sum_{i\in A_l, j\in A_l^C} \omega_{ij} - \frac{\gamma}{\mathrm{vol}_W(V)} \sum_{l=1}^{K} \sum_{i,j\in A_l} p_{ij}$$

$$= 1 - \frac{1}{\mathrm{vol}_W(V)} \left(\frac{1}{2} \sum_{l=1}^{K} \sum_{i\in A_l, j\in A_l^C} \omega_{ij} + \frac{1}{2} \sum_{l=1}^{K} \sum_{i\in A_l^C, j\in A_l} \omega_{ij} + \gamma \sum_{l=1}^{K} \sum_{i,j\in A_l} p_{ij} \right)$$

$$= 1 - \frac{1}{\mathrm{vol}_W(V)} \left(\frac{1}{2} \mathcal{TV}_W(U) + \frac{\gamma}{2} \mathcal{TV}_P^+(U) - \frac{\gamma}{2}(K-2)\mathrm{vol}_P(V) \right), \tag{27}$$

where in the last line $U \in Pt(K)$ corresponds to the partition $\mathcal{A}$.

Thus,

$$\mathcal{Q}_\gamma(\mathcal{A}; W, P) = -\frac{1}{\mathrm{vol}_W(V)} \mathcal{Q}_{\mathrm{mul},\gamma}(U; W, P) + 1 + \frac{\gamma(K-2)\mathrm{vol}_P(V)}{2\mathrm{vol}_W(V)},$$

if U corresponds to $\mathcal{A}$ and

$$\mathcal{Q}_{\mathrm{mul},\gamma}(U; W, P) := \frac{1}{2} \mathcal{TV}_W(U) + \frac{\gamma}{2} \mathcal{TV}_P^+(U).$$

Thus, the maximisation of $\mathcal{Q}_\gamma(\mathcal{A}; W, P)$ over all partitions $\mathcal{A}$ with fixed K is equivalent to the minimisation of $\mathcal{Q}_{\mathrm{mul},\gamma}(U; W, P)$ over all $U \in Pt(K)$. If $K = 2$ and U and v are related as in (24), then $\mathcal{Q}_{\mathrm{mul},\gamma}(U; W, P) = \mathcal{Q}_{\mathrm{bin},\gamma}(v; W, P)$.

Similar to what we did in the case of bipartitions in Section 3.1, if we split $B_\gamma = W - \gamma P$ as $B_\gamma = B_\gamma^+ - B_\gamma^-$, we can do the analogue computation to (27) to find

$$\mathcal{Q}_\gamma(\mathcal{A}; W, P) = \frac{1}{\mathrm{vol}_W(V)} \sum_{i,j\in V} \left((b_\gamma^+)_{ij} - (b_\gamma^-)_{ij} \right) \delta(c_i, c_j)$$

$$= -\frac{1}{\mathrm{vol}_W(V)} \mathcal{Q}_{mul,1}(U; B_\gamma^+, B_\gamma^-) + \frac{2\mathrm{vol}_{B_\gamma^+}(V) + (K-2)\mathrm{vol}_{B_\gamma^-}(V)}{2\mathrm{vol}_W(V)}.$$

Hence, we can also maximise $\mathcal{Q}_\gamma(\mathcal{A}; W, P)$ over all partitions $\mathcal{A}$ containing K subsets by minimising $\mathcal{Q}_{mul,1}(U; B_\gamma^+, B_\gamma^-)$ over all $U \in Pt(K)$. We emphasise that in $\mathcal{Q}_{mul,1}$ we choose $\gamma = 1$, since the influence of γ is now in the matrices B_γ^+ and B_γ^-, rather than in the structure of the function(al).

4. Diffuse-interface methods

Diffuse-interface methods[16] [4, 5] use efficient PDE techniques to handle segmentation problems. The GL functional associated with a graph Laplacian, whose minimisation is associated with the minimisation of the total variation, is widely used in diffuse-interface approaches.

4.1. Binary classification with graph Ginzburg–Landau functionals

A central object in the diffuse-interface approach of [4] is the graph GL functional $f_\varepsilon : \mathcal{V} \to \mathbb{R}$ defined by:

$$f_\varepsilon(u) := \frac{1}{4} \sum_{i,j\in V} \omega_{ij} \left(u_i - u_j\right)^2 + \frac{1}{\varepsilon} \sum_{i\in V} \Phi\left(u_i\right) = \frac{1}{2} \langle u, L_W u \rangle + \frac{1}{\varepsilon} \sum_{i\in V} \Phi\left(u_i\right) \qquad \text{(by (6))}$$

$$= \frac{1}{2} \langle u, L_{W_{\mathrm{rw}}} u \rangle_W + \frac{1}{\varepsilon} \sum_{i\in V} \Phi\left(u_i\right) \qquad \text{(by (8))}, \tag{28}$$

[16] The name hints at the origins of these methods in continuum models for phase separation.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

where $\varepsilon > 0$ is a parameter and the function $\Phi(u)$ is a double-well potential with two minima. For example, a classical choice is the polynomial $\Phi(u) = \frac{1}{4}(u^2 - 1)^2$ that has minima at $u = -1$ and $u = +1$. The first of the two terms in (28) is called the graph Dirichlet energy.

In Van Gennip and Bertozzi [72], it is proven that if $\varepsilon \downarrow 0$, then the sequence of functionals f_ε Γ-converges to

$$f_0(u) := \begin{cases} TV_W(u), & \text{if } u \in \mathcal{V}_{\text{bin}}, \\ +\infty, & \text{otherwise.} \end{cases}$$

For details about Γ-convergence, we refer to Dal Maso [47] and Braides [8]. Here, it suffices to note that Γ-convergence of f_ε combined with an equicoercivity condition, which is also satisfied in this case (see [72]), implies that minimisers of f_ε converge to minimisers of f_0 as $\varepsilon \downarrow 0$.

Similarly, in Keetch and Van Gennip [39], the signless graph GL functional $f_0^+ : \mathcal{V} \to \mathbb{R}$, defined as

$$f_\varepsilon^+(u) := \frac{1}{4}\sum_{i,j\in V} \omega_{ij}\left(u_i + u_j\right)^2 + \frac{1}{\varepsilon}\sum_{i\in V} \Phi\left(u_i\right) = \frac{1}{2}\langle u, Q_W u\rangle + \frac{1}{\varepsilon}\sum_{i\in V} \Phi\left(u_i\right) \qquad \text{(by (10))}$$

$$= \frac{1}{2}\langle u, Q_{W_{\text{rw}}} u\rangle_W + \frac{1}{\varepsilon}\sum_{i\in V} \Phi\left(u_i\right) \qquad \text{(by (12)),}$$

is introduced and it is proven that f_ε^+ Γ-converges to

$$f_0^+(u) := \begin{cases} TV_W^+(u), & \text{if } u \in \mathcal{V}_{\text{bin}}, \\ +\infty, & \text{otherwise,} \end{cases}$$

as $\varepsilon \downarrow 0$. Also in this case the equicoercivity condition that is required to conclude convergence of minimisers of f_ε^+ to minimisers of f_0^+ is satisfied (see [39]). We call the first of the two terms in f_0^+ the signless graph Dirichlet energy.

A straightforward adaptation of the proofs in [72, Theorems 3.1 and 3.2] and [39, Lemmas 4.3 and 4.4] shows that if we define

$$f_{\varepsilon,\gamma}^{\pm}(u; W, P) := \frac{1}{4}\sum_{i,j\in V} \omega_{ij}\left(u_i - u_j\right)^2 + \frac{\gamma}{4}\sum_{i,j\in V} p_{ij}\left(u_i + u_j\right)^2 + \frac{1}{\varepsilon}\sum_{i\in V} \Phi\left(u_i\right),$$

then $f_{\varepsilon,\gamma}^{\pm}(\,\cdot\,; W, P)$ Γ-converges to

$$f_{0,\gamma}^{\pm}(u; W, P) := \begin{cases} \mathcal{Q}_{\text{bin},\gamma}(u; W, P), & \text{if } u \in \mathcal{V}_{\text{bin}}, \\ +\infty, & \text{otherwise,} \end{cases}$$

and the required equicoercivity conditions are again satisfied that allow us to conclude that minimisers of $f_{\varepsilon,\gamma}^{\pm}(\,\cdot\,; W, P)$ converge to minimisers of $\mathcal{Q}_{\text{bin},\gamma}(\,\cdot\,; W, P)$ from (19). We provide more details about the proof when we consider the case with multiple clusters in Theorem 4.1.

Similarly, $f_{\varepsilon,\gamma}^{\pm}(\,\cdot\,; B_\gamma^+, B_\gamma^-)$ Γ-converges to $f_{0,\gamma}^{\pm}(u; B_\gamma^+, B_\gamma^-)$ and the required equicoercivity conditions are again satisfied.

For small ε, minimisers of $f_{\varepsilon,\gamma}^{\pm}(\,\cdot\,; W, P)$ (or $f_{\varepsilon,1}^{\pm}(\,\cdot\,; B_\gamma^+, B_\gamma^-)$) thus approximate (in the sense just described[17]) minimisers of $\mathcal{Q}_{\text{bin},\gamma}(\,\cdot\,; W, P)$ (or $\mathcal{Q}_{\text{bin},1}(\,\cdot\,; B_\gamma^+, B_\gamma^-)$), which we know to be equivalent to maximisers of modularity $\mathcal{Q}(\,\cdot\,; W, P)$ restricted to bipartitions of V.

Finding global minimisers of the non-convex function $f_\varepsilon^{\pm}$ is a difficult task. Instead, we can focus on finding local minimisers and hope that these, in practice, are also good approximations for maximisers of $\mathcal{Q}_\gamma$. To which extent this hope proves to be justified will be investigated numerically in Section 7. For now, we focus on the problem of finding such local minimisers.

[17] We do not have an error estimate on the quality of the approximation.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

One possible method is to compute a gradient flow of $f_\varepsilon^\pm$ (see, e.g., Van Gennip *et al.* [73] and Keetch and Van Gennip [39]). For $u, v \in \mathcal{V}$ and $s \in \mathbb{R}$, using the self-adjointness of L_W, $L_{W_{\mathrm{rw}}}$, Q_W, and $Q_{W_{\mathrm{rw}}}$, we compute[18]

$$\begin{aligned}\frac{d}{ds} f^\pm_{\varepsilon,\gamma}(u+sv; W, P)\bigg|_{s=0} &= \langle L_W u, v\rangle + \gamma \langle Q_P u, v\rangle + \frac{1}{\varepsilon}\langle \Phi' \circ u, v\rangle \\ &= \langle L_{W_{\mathrm{rw}}} u, v\rangle_W + \gamma \langle Q_{P_{\mathrm{rw}}} u, v\rangle_W + \frac{1}{\varepsilon}\langle D_W^{-1}\Phi' \circ u, v\rangle_W.\end{aligned}$$

Thus, the gradient flows of $f^\pm_{\varepsilon,\gamma}(\,\cdot\,; W, P)$ with respect to the Euclidean inner product and the degree-weighted inner product are

$$\frac{du}{dt} = -L_W u - \gamma Q_P u - \frac{1}{\varepsilon}\Phi' \circ u \quad \text{and} \quad \frac{du}{dt} = -L_{W_{\mathrm{rw}}} u - \gamma Q_{P_{\mathrm{rw}}} u - \frac{1}{\varepsilon} D_W^{-1}\Phi' \circ u, \tag{29}$$

respectively. As is standard for gradient flows, a dependence on 'time' t has been introduced so that we now may interpret u as a function $u : \mathbb{R} \to \mathcal{V}$.

In a completely analogous manner, we determine

$$\frac{d}{ds} f^\pm_{\varepsilon,1}(u+sv; B^+_\gamma, B^-_\gamma)\bigg|_{s=0} = \langle L_{B^+} u, v\rangle + \langle Q_{B^-_\gamma} u, v\rangle + \frac{1}{\varepsilon}\langle \Phi' \circ u, v\rangle$$

and thus the gradient flow of $f^\pm_{\varepsilon,1}(\,\cdot\,; B^+_\gamma, B^-_\gamma)$ with respect to the Euclidean inner product is

$$\frac{du}{dt} = -L_{B^+} u - Q_{B^-} u - \frac{1}{\varepsilon}\Phi' \circ u. \tag{30}$$

For the gradient flow with respect to a degree-weighted inner product, the situation is more complicated. We recall the definition of the degree-weighted inner product in (2) and explicitly will be using the following versions:

$$\langle u, v\rangle_{B_\gamma} = \sum_{i\in V} u_i v_i (d_{B_\gamma})_i, \quad \langle u, v\rangle_{B^+_\gamma} = \sum_{i\in V} u_i v_i (d_{B^+_\gamma})_i, \quad \text{and} \quad \langle u, v\rangle_{B^-_\gamma} = \sum_{i\in V} u_i v_i (d_{B^-_\gamma})_i. \tag{31}$$

By (20), we know that, if $D_W = D_P$ and $\gamma = 1$, then $D_{B_1} = 0$ and therefore $D_{B^+_1} = D_{B^-_1}$. Thus in this case, the B^+_γ- and B^-_γ-degree-weighted inner products are equal and the B_γ-degree-weighted inner product is always zero.[19] If additionally $D_{B^+_1} = D_{B^-_1}$ is invertible (e.g., because the remaining assumption from Corollary 3.2 part 2 is satisfied), then we can write

$$\langle L_{B^+_1} u, v\rangle = \langle L_{B^+_{1\,\mathrm{rw}}} u, v\rangle_{B^+_1}, \quad \langle Q_{B^-_1} u, v\rangle = \langle Q_{B^-_{1\,\mathrm{rw}}} u, v\rangle_{B^+_1}, \quad \text{and} \quad \langle \Phi'(u), v\rangle = \langle D^{-1}_{B^+_1}\Phi'(u), v\rangle_{B^+_1}.$$

Hence, if $\gamma = 1$, the gradient flow with respect to the B^+-degree-weighted inner product is

$$\frac{du}{dt} = -L_{B^+_{1\,\mathrm{rw}}} u - Q_{B^-_{1\,\mathrm{rw}}} u - \frac{1}{\varepsilon} D^{-1}_{B^+_1}\Phi' \circ u. \tag{32}$$

If $\gamma \neq 1$ and $D_W = D_P$, then the degree matrices with respect to B^+_γ and B^-_γ are no longer the same, but rather we have

$$D_{B^-_\gamma} = D_{B^+_\gamma} - D_{B_\gamma} = D_{B^+_\gamma} - (1-\gamma) D_W. \tag{33}$$

This means we have to make a choice to use either the B^+_γ- or B^-_γ-degree-weighted inner product, as they are no longer identical. That choice will influence the resulting equations. We choose the former; calculations for the alternative choice are similar.

With this choice, and still assuming that $D_{B^-_\gamma}$ and $D_{B^+_\gamma}$ are invertible, for example because the assumptions from Corollary 3.2 part 1 hold, we still have

$$\langle L_{B^+_\gamma} u, v\rangle = \langle L_{B^+_{\gamma\,\mathrm{rw}}} u, v\rangle_{B^+_\gamma} \quad \text{and} \quad \langle \Phi'(u), v\rangle = \langle D^{-1}_{B^+_\gamma}\Phi'(u), v\rangle_{B^+_\gamma},$$

[18] In a slight abuse of notation, we interpret $\Phi'(u)$ as the vector obtained by applying Φ elementwise to the vector (representation of) u so that the matrix-vector multiplication $D_W^{-1}\Phi'(u)$ is well defined.

[19] And thus strictly speaking is not an inner product.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

yet when rewriting the remaining term in $\frac{d}{ds} f^{\pm}_{\varepsilon,1}(u + sv; B^+_\gamma, B^-_\gamma)\big|_{s=0}$, the difference in B^+_γ- and B^-_γ-degree-weighted inner products is important:

$$\begin{aligned}\langle Q_{B^-_\gamma} u, v\rangle &= \langle Q_{B^-_{\gamma\,\mathrm{rw}}} u, v\rangle_{B^-_\gamma} = \langle Q_{B^-_{\gamma\,\mathrm{rw}}} u, v\rangle_{B^+_\gamma} - \langle Q_{B^-_{\gamma\,\mathrm{rw}}} u, v\rangle_{B_\gamma} \\ &= \langle Q_{B^-_{\gamma\,\mathrm{rw}}} u, v\rangle_{B^+_\gamma} - \langle D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\,\mathrm{rw}}} u, v\rangle_{B^+_\gamma}.\end{aligned}$$

To obtain the second equality above, we used the first equality in (33). For the first and third equalities, we used (3), with $C = I$, $\tilde{C} = B^-_\gamma$, and $C = B_\gamma$, $\tilde{C} = B^+_\gamma$, respectively. Thus, under the assumption that $D_W = D_P$ and that $D_{B^+_\gamma}$ and $D_{B^-_\gamma}$ are invertible, the gradient flow of $f^{\pm}_{\varepsilon,1}(\,\cdot\,; B^+_\gamma, B^-_\gamma)$ with respect to the B^+_γ-degree-weighted inner product is

$$\begin{aligned}\frac{du}{dt} &= -L_{B^+_{\gamma\,\mathrm{rw}}} u - Q_{B^-_{\gamma\,\mathrm{rw}}} u + D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\,\mathrm{rw}}} u - \frac{1}{\varepsilon} D^{-1}_{B^+_1} \Phi' \circ u \\ &= -L_{B^+_{\gamma\,\mathrm{rw}}} u - Q_{B^-_{\gamma\,\mathrm{rw}}} u + (1-\gamma) D^{-1}_{B^+_\gamma} D_W Q_{B^-_{\gamma\,\mathrm{rw}}} u - \frac{1}{\varepsilon} D^{-1}_{B^+_1} \Phi' \circ u.\end{aligned}$$

Here, we used (20) again. We note that this gradient flow indeed equals the one from (32) if $\gamma = 1$.

In fact, in this paper we do not solve these Allen–Cahn-type equations directly, which could be accomplished, for example, by a convex–concave splitting technique as in [4] (see Luo and Bertozzi [46] for an analysis of the scheme), but we use a related MBO scheme, which we introduce in Section 5. Before doing that, we first consider an extension of the graph GL functional to multiple clusters.

4.2. Multiclass clustering with graph Ginzburg–Landau functionals

The previous sections dealt with partitioning of the node set V into (at most) two subsets; now we turn our attention to the case where we allow partitions of up to and including K subsets, where $K \geq 2$ is fixed. We recall that we allow empty subsets in the partition. We base our approach on the method described in Garcia-Cardona *et al.* [24] and Merkurjev *et al.* [48].

To generalise the GL functional $f^{\pm}_{\varepsilon,\gamma}$ to the multiclass context, we require multiclass generalisations of the (signless) graph Dirichlet energies and of the double-well potential. For the Dirichlet energy, we generalise the term $\frac{1}{2}\langle u, L_W u\rangle$ to

$$\frac{1}{2}\langle U, L_W U\rangle = \frac{1}{2}\sum_{k=1}^{K} \langle U_{*k}, L_W U_{*k}\rangle = \frac{1}{4}\sum_{k=1}^{K}\sum_{i,j\in V} \omega_{ij}(U_{ik} - U_{jk})^2,$$

where, in a slight overload of the inner product notation, for matrices $U, V \in \mathbb{R}^{|V|\times K}$ we have defined

$$\langle U, V\rangle := \sum_{k=1}^{K} \langle U_{*k}, V_{*k}\rangle. \tag{34}$$

Similarly, we extend the W-degree-weighted inner product to matrices $U, V \in \mathbb{R}^{|V|\times K}$ by:

$$\langle U, V\rangle_W := \sum_{k=1}^{K} \langle U_{*k}, V_{*k}\rangle_W. \tag{35}$$

Hence,

$$\frac{1}{2}\langle U, L_W U\rangle = \frac{1}{2}\langle U, L_{W_{\mathrm{rw}}} U\rangle_W.$$

In a similar way, the signless graph Dirichlet energy $\frac{1}{2}\langle u, Q_W u\rangle$ is generalised to

$$\frac{1}{2}\langle U, Q_P U\rangle = \frac{1}{4}\sum_{k=1}^{K}\sum_{i,j\in V} p_{ij}(U_{ik} + U_{jk})^2 = \frac{1}{2}\langle U, Q_{P_{\mathrm{rw}}} U\rangle_P.$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

To generalise the double-well potential to a multiple-well potential, we recall from Section 3.2 that a partition of V into K subsets can be described by a matrix $U \in Pt(K)$. If we write U_{i*} for the i^{th} row of U, then $U_{i*} \in \{-1, 1\}^K$ as row vector and there exists a unique $k \in \{1, \dots, K\}$ such that $U_{ik} = 1$. We introduce a notation for such vectors. For $k \in \{1, \dots, K\}$, let $e^{(k)} \in \{-1, 1\}^K$ be the row vector that satisfies $e^{(k)}_k = 1$ and, for all $l \in \{1, \dots, K\} \setminus \{k\}$, $e^{(k)}_l = -1$. Now we define the multiple-well potential for vectors $w \in \mathbb{R}^K$ by:

$$\Phi_{\text{mul}}(w) := \frac{1}{2}\left(\prod_{k=1}^{K} \frac{1}{4}||w - e^{(k)}||_1^2\right). \tag{36}$$

Given a matrix $U \in \mathbb{R}^{V \times |K|}$, $\sum_{i \in V} \Phi_{\text{mul}}(U_{i*})$ is non-negative, and it is zero if and only if, for all $i \in V$, there exists a $k \in \{1, \dots, K\}$ such that $U_{i*} = e^{(k)}$. In other words, $U \mapsto \sum_{i \in V} \Phi_{\text{mul}}(U_{i*})$ achieves its global minimum exactly at each element of $Pt(K)$ and thus generalises the double-well potential term $\sum_{i \in V} \Phi(u)$, which achieves its global minimum exactly at all elements of $\mathcal{V}_{\text{bin}}$.

This brings us to the following multiclass variant of the functional $f^{\pm}_{\varepsilon,\gamma}(\,\cdot\,; W, P)$ for matrices $U \in \mathbb{R}^{|V| \times K}$:

$$\begin{aligned}
\mathcal{F}^{\pm}_{\varepsilon,\gamma}(U; W, P) &:= \frac{1}{8}\sum_{k=1}^{K}\sum_{i,j \in V} \omega_{ij}(U_{ik} - U_{jk})^2 + \frac{\gamma}{8}\sum_{k=1}^{K}\sum_{i,j \in V} p_{ij}(U_{ik} + U_{jk})^2 + \frac{1}{\varepsilon}\sum_{i \in V} \Phi_{\text{mul}}(U_{i*}) \\
&= \frac{1}{4}\langle U, L_W U\rangle + \frac{\gamma}{4}\langle U, Q_P U\rangle + \frac{1}{\varepsilon}\sum_{i \in V} \Phi_{\text{mul}}(U_{i*}) \\
&= \frac{1}{4}\langle U, L_{W_{\text{rw}}} U\rangle_W + \frac{\gamma}{4}\langle U, Q_{P_{\text{rw}}} U\rangle_P + \frac{1}{\varepsilon}\sum_{i \in V} \Phi_{\text{mul}}(U_{i*}).
\end{aligned}$$

As in the case of binary classification, we have a Γ-convergence result for $\mathcal{F}^{\pm}_{\varepsilon}$.

Theorem 4.1. *Let $K \in \mathbb{N}$ with $K \geq 2$ and $\gamma \in (0, \infty)$. Define $\mathcal{F}^{\pm}_0 : \mathbb{R}^{|V| \times K} \to \mathbb{R} \cup \{+\infty\}$ by*

$$\mathcal{F}^{\pm}_{0,\gamma}(U; W, P) := \begin{cases} \mathcal{Q}_{mul,\gamma}(U; W, P), & \text{if } U \in Pt(K), \\ +\infty, & \text{otherwise.} \end{cases}$$

Then $\mathcal{F}^{\pm}_{\varepsilon,\gamma}(\,\cdot\,; W, P)$ Γ-converges[20] to $\mathcal{F}^{\pm}_{0,\gamma}(\,\cdot\,; W, P)$ as $\varepsilon \downarrow 0$.

Moreover, if $(\varepsilon_n) \subset (0, \infty)$ is a sequence such that $\varepsilon_n \downarrow 0$ as $n \to \infty$ and $(U_n) \subset Pt(K)$ is a sequence for which there exists a $C > 0$ such that, for all $n \in \mathbb{N}$, $\mathcal{F}^{\pm}_{\varepsilon_n,\gamma}(U_n; W, P) \leq C$, then there exists a converging subsequence of (U_n) with limit in $Pt(K)$.

Proof. Our proof largely follows Van Gennip and Bertozzi [72] and Boyd *et al.* [7]. The strategy to prove Γ-convergence for $\mathcal{F}^{\pm}_{\varepsilon}$ is to first prove Γ-convergence of the functional $\varphi_\varepsilon(U) := \frac{1}{\varepsilon}\sum_{i=1}^{|V|} \Phi_{\text{mul}}(U_{i*})$ to

$$\varphi_0(U) := \begin{cases} 0, & \text{if } U \in Pt(K), \\ +\infty, & \text{if } U \in \mathbb{R}^{|V| \times K} \setminus Pt(K), \end{cases}$$

and then use the fact that Γ-convergence is preserved under addition of continuous terms (see Dal Maso [47] or Braides [8]) such as the (signless) graph Dirichlet terms. The proof then concludes with the observation that by (25) and (26), for all $U \in Pt(K)$,

$$\frac{1}{8}\sum_{k=1}^{K}\sum_{i,j \in V} \omega_{ij}(U_{ik} - U_{jk})^2 = \frac{1}{4}\sum_{k=1}^{K}\sum_{i,j \in V} \omega_{ij}|U_{ik} - U_{jk}| = \frac{1}{2}\mathcal{TV}_W(U)$$

[20] Since all norms on finite-dimensional vector spaces are equivalent, we do not need to specify which normed topology we consider for our Γ-convergence.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

and

$$\frac{1}{8}\sum_{k=1}^{K}\sum_{i,j\in V} p_{ij}(U_{ik}+U_{jk})^2 = \frac{1}{4}\sum_{k=1}^{K}\sum_{i,j\in V} p_{ij}|U_{ik}-U_{jk}| = \frac{1}{2}\mathcal{TV}_P^+(U).$$

Per definition, to establish Γ-convergence of φ_ε to φ_0, we have to prove two statements:

- **Lower bound.** For all sequences $(\varepsilon_n)\subset(0,\infty)$ that converge to zero, for all $U\in\mathbb{R}^{|V|\times K}$, and for all sequences $(U_n)\subset\mathbb{R}^{|V|\times K}$ that converge to U, it holds that
$$\varphi_0(U)\le\liminf_{n\to\infty}\varphi_{\varepsilon_n}(U_n).$$
- **Recovery sequence.** For all sequences $(\varepsilon_n)\subset(0,\infty)$ that converge to zero and for all $U\in\mathbb{R}^{|V|\times K}$, there exists a sequence $(U_n)\subset\mathbb{R}^{|V|\times K}$ that converges to U and such that
$$\varphi_0(U)\ge\limsup_{n\to\infty}\varphi_{\varepsilon_n}(U_n).$$

Let (ε_n) be a positive sequence such that $\varepsilon_n\downarrow 0$ as $n\to\infty$ and let $U\in\mathbb{R}^{|V|\times K}$.

To prove the lower bound condition, first we assume that $U\in Pt(K)$. Since, for all $n\in\mathbb{N}$, φ_{ε_n} is non-negative, we find that, for all $n\in\mathbb{N}$

$$\varphi_0(U)=0\le\varphi_{\varepsilon_n}(U_n)$$

and the required lim inf-inequality is satisfied.

If $U\in\mathbb{R}^{|V|\times K}\setminus Pt(K)$, then $\varphi_0(U)=+\infty$. Moreover, there exists a $i\in V$ such that, for all $k\in\{1,\dots,K\}$, $U_{i*}\neq e^{(k)}$. Since (U_{ε_n}) converges to U, it follows that there exists a radius $r>0$ such that for n large enough $(U_{\varepsilon_n})_{i*}\in\mathbb{R}^K\setminus\bigcup_{k=1}^K B(e^{(k)},r)$, where $B(e^{(k)},r)$ denotes the open ball with respect to the 1-norm in $\mathbb{R}^K$ centred at $e^{(k)}$ with radius r. This in turn implies that there exists a $\tilde{C}>0$ such that, for n large enough, $\Phi_{\mathrm{mul}}(U_{i*})\ge\tilde{C}$. Hence,

$$\liminf_{n\to\infty}\frac{1}{\varepsilon_n}\sum_{i\in V}\Phi_{\mathrm{mul}}(U_{i*})\ge\liminf_{n\to\infty}\frac{\tilde{C}}{\varepsilon_n}=+\infty,$$

thus,

$$\varphi_0(U)=+\infty=\liminf_{n\to\infty}\varphi_{\varepsilon_n}(U_n).$$

To prove existence of a recovery sequence, we note that the lim sup-inequality is trivially true if $U\in\mathbb{R}^{|V|\times K}\setminus Pt(K)$; hence, we assume that $U\in Pt(K)$. Let (U_n) be the constant sequence with, for all $n\in\mathbb{N}$, $U_n=U$. Then, for all $n\in\mathbb{N}$, $\varphi_{\varepsilon_n}(U_n)=0$. Since φ_0 is non-negative, the sequence (U_n) is indeed a recovery sequence.

Thus, φ_ε Γ-converges to φ_0 and, by our argument above, $\mathcal{F}_\varepsilon^\pm$ Γ-converges to $\mathcal{F}_0^\pm$.

To prove the equicoercivity statement in the second part of the theorem, we assume that $(\varepsilon_n)\subset(0,\infty)$ is a sequence such that $\varepsilon_n\downarrow 0$ as $n\to\infty$ and $(U_n)\subset Pt(K)$ is a sequence for which there exists a $C>0$ such that, for all $n\in\mathbb{N}$, $\mathcal{F}_{\varepsilon_n}^\pm(U_n)\le C$. In particular, this implies that for all $n\in\mathcal{N}$ and for all $i\in V$,

$$0\le\Phi_{\mathrm{mul}}((U_n)_{i*})\le C. \tag{37}$$

Let $i\in V$. If there exists a $k\in\{1,\dots,K\}$ such that the sequence $\left(\|(U_n)_{i*}-e^{(k)}\|_1\right)$ is unbounded in $\mathbb{R}$, then there must also exist an $l\in\{1,\dots,K\}$ such that the sequence $\left(\|(U_n)_{i*}-e^{(l)}\|_1\right)$ converges to zero; otherwise, the sequence $(\Phi_{\mathrm{mul}}((U_n)_{i*}))$ would be unbounded. This is a contradiction; hence, the sequence $((U_n)_{i*})$ is contained in a bounded subset of $\mathbb{R}^K$ and thus the sequence (U_n) is bounded in $\mathbb{R}^{|V|\times K}$. By Bolzano–Weierstraß, this sequence has a converging subsequence. Denote its limit by U_∞.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

If $U_\infty \in \mathbb{R}^{|V|\times K} \setminus Pt(K)$, then by the same argument as in the proof of the lower bound, we know that

$$\liminf_{n\to\infty} \frac{1}{\varepsilon_n} \sum_{i\in V} \Phi_{\text{mul}}(U_{i*}) = +\infty,$$

which contradicts (37). Hence, $U_\infty \in Pt(K)$, which completes the proof. □

As discussed in Section 4.1, the Γ-convergence and equicoercivity results of Theorem 4.1 imply that minimisers of $\mathcal{F}_\varepsilon^\pm$ converge to minimisers of $\mathcal{F}_0^\pm$ as $\varepsilon \downarrow 0$.

In a completely analogous way to what we did at the end of Section 4.1, we may now derive gradient flows of $\mathcal{F}_\varepsilon^\pm$ with respect to the Euclidean or W-degree-weighted inner product for matrices. Because we will not actually use these gradient flows to find minimisers, but use MBO schemes (see Section 5.3) instead, we leave out the details of the derivation and only mention that formally[21] we recover, for all $k \in \{1, \dots, K\}$, the Allen–Cahn-type equations:

$$\frac{dU_{*k}}{dt} = -L_W U_{*k} - Q_P U_{*k} - \frac{1}{\varepsilon}(\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k} \quad \text{and} \quad \frac{dU_{*k}}{dt} = -L_{W_{\text{rw}}} U_{*k} - Q_{P_{\text{rw}}} U_{*k} - \frac{1}{\varepsilon}(D_W^{-1}\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k},$$

where $(\mathcal{D}\Phi_{\text{mul}} \circ U)_{ik} := \partial_k \Phi_{\text{mul}}(U_{i*})$, with ∂_k the partial derivative operator with respect to the k^{th} variable.

Remark 4.2. As in Section 4.1, we can recover similar results as above, if we consider the functionals $\mathcal{F}_{\varepsilon,1}^\pm(\,\cdot\,; B_\gamma^+, B_\gamma^-)$. Their Γ-limit for $\varepsilon \downarrow 0$ is $\mathcal{F}_{0,1}^\pm(\,\cdot\,; B_\gamma^+, B_\gamma^-)$ and their gradient flow with respect to the Euclidean inner product is, for all $k \in \{1, \dots, L\}$,

$$\frac{dU_{*k}}{dt} = -L_{B^+} U_{*k} - Q_{B^-} U_{*k} - \frac{1}{\varepsilon}(\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k}.$$

The gradient flow with respect to the B_γ^+-degree-weighted inner product is, for all $k \in \{1, \dots, L\}$,

$$\begin{aligned}\frac{dU_{*k}}{dt} &= -L_{B^+_{\text{rw}}} U_{*k} - Q_{B^-_{\text{rw}}} U_{*k} + D_{B_\gamma^+}^{-1} D_{B_\gamma} Q_{B^-_{\gamma\,\text{rw}}} u - \frac{1}{\varepsilon}(D_{B^+}^{-1}\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k} \\ &= -L_{B^+_{\text{rw}}} U_{*k} - Q_{B^-_{\text{rw}}} U_{*k} + (1-\gamma) D_{B_\gamma^+}^{-1} D_W Q_{B^-_{\gamma\,\text{rw}}} u - \frac{1}{\varepsilon}(D_{B^+}^{-1}\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k}.\end{aligned}$$

We recall that we have used $D_W = D_P$ and invertibility of $D_{B_\gamma^+}$ and of $D_{B_\gamma^-}$ (as is guaranteed if assumptions of Corollary 3.2, part 1 are satisfied).

[21] Because of the 1-norms in its definition (36), Φ_{mul} is not differentiable at its wells, that is, at the vertices of the simplex $\mathfrak{S}(K)$ from (44). Instead of Allen–Cahn-type differential equations, we may consider differential *inclusions* such as $\frac{dU_{*k}}{dt} \in -L_W U_{*k} - Q_P U_{*k} - \frac{1}{\varepsilon}(\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k}$, and similarly in the second case, where now $(\mathcal{D}\Phi_{\text{mul}} \circ U)_{*k}$ denotes the Clarke subdifferential of Φ_{mul} at U_{*k}.

Since Φ_{mul} is not convex, we cannot use the standard subdifferential. The Clarke subdifferential is an extension of the standard subdifferential, which is well defined for locally Lipschitz-continuous functions. It has many of the same useful properties that the standard subdifferential has. For example, for convex lower semicontinuous functions, the Clarke subdifferential is equal to the standard subdifferential; the Clarke subdifferential of a function at a point where the function is Fréchet differentiable is the singleton containing the Fréchet derivative at that point; and the Clarke subdifferential at a point at which the function has a local minimum contains 0. For the definition of the Clarke subdifferential (also called the generalised gradient) and the proofs of these and other properties, see, for example, Clarke [17, Chapter 2] and Clason [18, Section 8].

Since the 1-norm is locally Lipschitz continuous at every point in its domain and the product of locally Lipschitz continuous functions is also locally Lipschitz continuous, the function Φ_{mul} is locally Lipschitz on its domain. Thus, its Clarke subdifferential is well defined.

Other ways of defining gradient flows, such as variational inequalities, may be considered as well, but it goes far beyond the scope of the current paper to investigate these options; see Ambrosio *et al.* [1].

One may wonder if all these difficulties could not be avoided by replacing the 1-norms in Φ_{mul} by 2-norms. Indeed, the results from Theorem 4.1 would then still hold, with little to no changes to the proof. In Merkurjev *et al.* [48, Section 2.1] and Garcia-Cardona *et al.* [24, Section 3.1], it is stated the 1-norms are used to avoid Φ_{mul} having a (local) minimiser in the middle of the simplex $\mathfrak{S}(K)$. We provide details in Appendix A.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

We also note that the Γ-convergence and equicoercivity proofs in Theorem 4.1 mostly depended on the potential term and can thus easily be reproduced with the matrices B^+ and B^- replacing W and P, respectively.

Remark 4.3. So far, we have ignored the question of existence of minimisers, but it is worth considering. The sets $\mathcal{V}_{\text{bin}}$ and $Pt(K)$ are finite sets and hence the functionals $f_{0,\gamma}$, $f^+_{0,\gamma}$, $f^\pm_{0,\gamma}$ and $\mathcal{F}^\pm_{0,\gamma}$ trivially have minimisers.

On the other hand, the functionals $f_{\varepsilon,\gamma}$, $f^+_{\varepsilon,\gamma}$ and $f^\pm_{\varepsilon,\gamma}$ are all continuous on $\mathcal{V}$ (or $\mathbb{R}^{|V|}$), and $\mathcal{F}^\pm_{\varepsilon,\gamma}$ is continuous on $\mathbb{R}^{|V|\times K}$. All these functionals are also bounded below (by zero) and the coercivity of the double-well or multiple-well potential terms allows us to restrict minimising sequences to a compact subset of $\mathbb{R}^{|V|}$ or $\mathbb{R}^{|V|\times K}$, in a similar way to what was done in the equicoercivity proof in Theorem 4.1. Hence, minimisers of these functionals exist.

5. MBO schemes

The MBO scheme was originally introduced by Merriman *et al.* [50, 51] as an algorithm for producing flow by mean curvature. It is a simple yet powerful iterative scheme which alternates between diffusion and thresholding. There is a large body of literature dealing with applications and computational and theoretical aspects of this scheme. For brevity, we focus on the use of the scheme for modularity optimisation.

5.1. MBO schemes for binary community detection

The MBO scheme was adapted to graphs by Merkurjev *et al.* [49] with the goal of (approximately) minimising the GL functional f_ε. It consists of iteratively performing short-time graph diffusion by solving $\frac{du}{dt} = -Lu$, followed by a hard thresholding step to mimic the drive towards the wells of Φ from the non-linear reaction term in the gradient flow. Here, we take $L \in \{L_W, L_{W_{\text{rw}}}, L_{W_{\text{sym}}}\}$ to allow for different variants of the scheme. An MBO-type scheme is also employed by Hu *et al.* in [33] and Boyd *et al.* in [7] for modularity optimisation.

Unless specified differently, we assume that

$$L_{\text{mix}} \in \Big\{ L_W + \gamma Q_P, L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}, L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}, \\ L_{B^+_\gamma} + Q_{B^-_\gamma}, L_{B^+_{\gamma\,\text{sym}}} + Q_{B^-_{\gamma\,\text{sym}}}, L_{B^+_{\gamma\,\text{rw}}} + Q_{B^-_{\gamma\,\text{rw}}} - D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\,\text{rw}}} \Big\}, \tag{38}$$

This assumption is motivated by the linear operators that appear in the gradient flows of Sections 4.1 and 4.2.

For later reference, we note that

$$I - D^{-1}_{B^+_\gamma} D_{B_\gamma} = I - D^{-1}_{B^+_\gamma}(D_{B^+_\gamma} - D_{B^-_\gamma}) = D^{-1}_{B^+_\gamma} D_{B^-_\gamma}$$

and thus for the last choice of L_{mix} in (38), we can also write

$$L_{B^+_{\gamma\,\text{rw}}} + Q_{B^-_{\gamma\,\text{rw}}} - D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\,\text{rw}}} = L_{B^+_{\gamma\,\text{rw}}} + (I - D^{-1}_{B^+_\gamma} D_{B_\gamma}) Q_{B^-_{\gamma\,\text{rw}}} = L_{B^+_{\gamma\,\text{rw}}} + D^{-1}_{B^+_\gamma} D_{B^-_\gamma} Q_{B^-_{\gamma\,\text{rw}}} \\ = L_{B^+_{\gamma\,\text{rw}}} + D^{-1}_{B^+_\gamma} Q_{B^-_\gamma}. \tag{39}$$

This form is often easier to work with, but it hides partly the fact that it is the random walk variant of L_{mix} for the split of B_γ into a positive and negative part.

We recall from Section 2.2 that connectedness of G implies invertibility of D_W and, if the null model is such that $D_W = D_P$, also the invertibility of D_P, so that the choices of L_{mix} that involve inverses of those matrices are well defined. For the choices that require inverses of $D_{B^+_\gamma}$ or $D_{B^-_\gamma}$ we need the additional assumption that these matrices are invertible, which holds, for example, in the situation of Corollary 3.2.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Adapting the idea in [49] to the equations in (29), we propose the following MBO-type scheme.

Binary L_{mix}modularity MBO scheme

- **Initialise.** Choose an initial condition $u^0 \in \mathcal{V}_{\text{bin}}$, a 'time step' $\tau > 0$, and L_{mix} as in (38).
- **Step $n+1$: linear dynamics.** Solve the equation $\frac{du}{dt} = -L_{\text{mix}}u$ on $(0, \tau]$ with initial condition $u(0) = u^n$.
- **Step $n+1$: threshold.** Define, for all $i \in V$, $u_i^{n+1} := \begin{cases} -1, & \text{if } u_i(\tau) < 0, \\ 1, & \text{if } u_i(\tau) \geq 0. \end{cases}$
- **Stop.** Stop the scheme when a stopping condition or predetermined number of steps has been achieved.

To indicate explicitly the dependence on the choice of L_{mix}, we call this the binary L_{mix} modularity MBO (MMBO) scheme. We note that, besides the choices of L_{mix} that follow from equations (29), (30), and (32), we also allow the variant choices[22] $L_{\text{mix}} \in \{L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}, L_{B^+_{\gamma\,\text{sym}}} + Q_{B^-_{\gamma\,\text{sym}}}\}$.

We briefly mention here that we will choose the value for τ in the MMBO scheme via the method presented in [7]. It is an effective strategy and requires less manual adjustment of τ than other approaches. We give more details in Section 6.1.

5.2. Numerical schemes for binary MMBO

We employ two different numerical methods to implement the binary MMBO scheme, where the difference is found in how the methods solve the linear-dynamics step. One method uses (a truncation of) the closed-form solution [29, 30], while the other uses an implicit finite-difference Euler scheme [14]. In both methods, we need (the leading) eigenvalues and eigenvectors of L_{mix}.

5.2.1. Closed-form matrix exponential solution

A closed-form solution at $t = \tau$ of the equation $\frac{du}{dt} = -L_{\text{mix}}u$ with initial condition $u(0) = u^n$ is given by [29]:

$$u(\tau) = e^{-\tau L_{\text{mix}}} u^n.$$

Here, $e^{-\tau L_{\text{mix}}}$ is the matrix exponential, which is defined by its series expansion:

$$e^{-\tau L_{\text{mix}}} := I + \sum_{k=1}^{\infty} \frac{1}{k!}(-\tau L_{\text{mix}})^k.$$

[22] In [10, Theorem 4.1.2], it is shown, in a generalised setting, that $L_{W_{\text{sym}}}u$ would appear in the gradient flow, if the functional of which the flow is taken (with respect to the Euclidean inner product) has a Dirichlet-type term of the form:

$$\frac{1}{4}\sum_{i,j\in V} \omega_{ij}\left((d_W)_i^{-\frac{1}{2}}u_i - (d_W)_j^{-\frac{1}{2}}u_j\right)^2.$$

Similar calculations can be performed for $Q_{P_{\text{sym}}}$, $L_{B^+_{\text{sym}}}$ and $Q_{B^-_{\text{sym}}}$. Such Dirichlet terms do not fit well in our setting, because calculations as the ones in (17) that depend on the binary character of u_i cannot be replicated for $(d_W)_i^{-\frac{1}{2}}u_i$. Despite this, we still consider choices of L_{mix} based on these symmetrically normalised operators in our numerical tests and compare their performance with that of the other choices.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

If L_{mix} has real eigenvalues $\lambda_1 \leq \ldots \leq \lambda_{|V|}$ with corresponding, linearly independent, eigenvectors ξ_i, $i \in \{1, \ldots, |V|\}$, then $e^{-\tau L_{\text{mix}}}$ has eigenvalues $e^{-\tau\lambda_i}$ with the same eigenvectors. Hence,

$$u(\tau) = \sum_{i=1}^{|V|} c_i e^{-\tau\lambda_i} \xi_i, \tag{40}$$

for appropriate coefficients $c_i \in \mathbb{R}$.

The following lemma shows that for each of our choices, L_{mix} is indeed diagonalisable and thus has $|V|$ linearly independent eigenvectors.

Lemma 5.1. *Assume L_{mix} satisfies (38) and D_P is invertible, where this is needed for L_{mix} to be well defined. Then L_{mix} has $|V|$ (possibly repeated) non-negative real eigenvalues with corresponding linearly independent normalised eigenvectors, where the normalisation is specified in each of the cases below.*

Denote by X a matrix having these eigenvectors as columns (in any order) and by Λ, the diagonal matrix containing the corresponding eigenvalues in the same order. Then L_{mix} is real diagonalisable (in the standard Euclidean inner product structure), that is, $L_{mix} = X\Lambda X^{-1}$.

(a) Let $L_{mix} \in \{L_W + \gamma Q_P, L_{W_{sym}} + \gamma Q_{P_{sym}}, L_{B^+_\gamma} + Q_{B^-_\gamma}, L_{B^+_{\gamma\ sym}} + Q_{B^-_{\gamma\ sym}}\}$. In the case that $L_{mix} = L_{B^+_{\gamma\ sym}} + Q_{B^-_{\gamma\ sym}}$, assume that $D_{B^+_\gamma}$ and $D_{B^-_\gamma}$ are invertible. Take for the columns of X eigenvectors of L_{mix} with unit Euclidean norm. Then X is orthogonal, that is, $X^{-1} = X^T$.

(b) Let $L_{mix} = L_{W_{rw}} + \gamma Q_{P_{rw}}$. Assume that the null model is such that $D_P = D_W$. Moreover, take for the columns of X eigenvectors of L_{mix} with unit norm with respect to the W-degree-weighted inner product from (2). Then $X^{-1} = \tilde{X}^T D_W^{\frac{1}{2}}$, where $\tilde{X}$ is the orthogonal matrix containing the Euclidean-normalised eigenvectors of $L_{W_{sym}} + \gamma Q_{P_{sym}}$ as columns, in the same order as the eigenvalues in Λ.

(c) Assume that $D_{B^+_\gamma}$ and $D_{B^-_\gamma}$ are invertible. Let $L_{mix} = L_{B^+_{\gamma\ rw}} + Q_{B^-_{\gamma\ rw}} - D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\ rw}}$. Moreover, take for the columns of X eigenvectors of L_{mix} with unit norm with respect to the B^+_γ-degree-weighted inner product from (31). Then $X^{-1} = \tilde{X}^T D^{\frac{1}{2}}_{B^+_\gamma}$, where $\tilde{X}$ is the orthogonal matrix containing the Euclidean-normalised eigenvectors of $L_{B^+_{\gamma\ sym}} + D^{-\frac{1}{2}}_{B^+_\gamma} Q_{B^-_\gamma} D^{-\frac{1}{2}}_{B^+_\gamma}$ as columns, in the same order as the eigenvalues in Λ.

Proof. The proof is given in Appendix B. □

Remark 5.2. We note that the proof of Lemma 5.1, part (b) also establishes that $L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$ and $L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$ have the same eigenvalues. Moreover, the proof of Lemma 5.1, part (c) establishes that $L_{B^+_{\text{rw}}} + Q_{B^-_{\text{rw}}} - D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\ \text{rw}}}$ and $L_{B^+_{\gamma\ \text{sym}}} + D^{-\frac{1}{2}}_{B^+_\gamma} Q_{B^-_\gamma} D^{-\frac{1}{2}}_{B^+_\gamma}$ have the same eigenvalues.

Remark 5.3. If $\gamma = 1$ and $D_W = D_P$ in case (c) of Lemma 5.1, the situation simplifies, since then $D_{B_1} = 0$ and $D_{B^+_1} = D_{B^-_1}$ and thus

$$D^{\frac{1}{2}}_{B^+_1} L_{\text{mix}} D^{-\frac{1}{2}}_{B^+_1} = D^{\frac{1}{2}}_{B^+_1} \left(L_{B^+_{1\ \text{rw}}} + Q_{B^-_{1\ \text{rw}}} \right) D^{-\frac{1}{2}}_{B^+_1} = L_{B^+_{1\ \text{sym}}} + Q_{B^-_{1\ \text{sym}}}.$$

Part (a) of Lemma 5.1 then implies that L_{mix} is real diagonalisable and $X = \tilde{D}^{-\frac{1}{2}}_{B^+_1} X$, where $\tilde{X}$ is the matrix of Euclidean-normalised eigenvectors of $L_{B^+_{1\ \text{sym}}} + Q_{B^-_{1\ \text{sym}}}$. Moreover, L_{mix} has non-negative eigenvalues since both $L_{B^+_{1\ \text{sym}}}$ and $Q_{B^-_{1\ \text{sym}}}$, and thus also their sum, are positive semidefinite with respect to the Euclidean inner product by Lemma 2.1 part (b) and Lemma 2.2 part (b). We note that this result is consistent with the result in part (c) of Lemma 5.1, since, in this case, $L_{B^+_{1\ \text{sym}}} + D^{-\frac{1}{2}}_{B^+_1} Q_{B^-_1} D^{-\frac{1}{2}}_{B^+_1} = L_{B^+_{1\ \text{sym}}} + Q_{B^-_{1\ \text{sym}}}$.

Remark 5.4. In case (b) of Lemma 5.1, we know that L_{mix} is self-adjoint with respect to the inner product $\langle \cdot, \cdot \rangle_W$ (see Section 2.2) and thus we expect the columns of X to be orthonormal with respect to

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

this inner product. Indeed, we compute

$$X^T D_W X = \tilde{X}^T D_W^{-\frac{1}{2}} D_W^{\frac{1}{2}} D_W^{-\frac{1}{2}} \tilde{X} = \tilde{X}^T \tilde{X} = I.$$

A similar conclusion holds in case (c) of the lemma, yet now with respect to the B_γ^+-degree-weighted inner product from (31) as can easily be seen by replacing W by B_γ^+ in the computation above. This is not surprising, as we know from the proof of the lemma, that $D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}}$ is symmetric and thus is self-adjoint with respect to the Euclidean inner product; hence, L_{mix} is self-adjoint with respect to the B_γ^+-degree-weighted inner product, since, for all $u, v \in \mathcal{V}$,

$$\begin{aligned}\langle L_{\text{mix}} u, v\rangle_{B_\gamma^+} &= \langle D_{B_\gamma^+} L_{\text{mix}} u, v\rangle = \langle D_{B_\gamma^+}^{\frac{1}{2}} (D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}})(D_{B_\gamma^+}^{\frac{1}{2}} u), v\rangle \\ &= \langle D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}} (D_{B_\gamma^+}^{\frac{1}{2}} u), D_{B_\gamma^+}^{\frac{1}{2}} v\rangle = \langle D_{B_\gamma^+}^{\frac{1}{2}} u, D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}} (D_{B_\gamma^+}^{\frac{1}{2}} v)\rangle = \langle u, L_{\text{mix}} v\rangle_{B_\gamma^+}.\end{aligned}$$

For the next lemma, we recall that $\gamma > 0$.

Lemma 5.5.

(a) Let $L_{mix} \in \{L_W + \gamma Q_P, L_{W_{sym}} + \gamma Q_{P_{sym}}\}$. *Assume* D_P *is invertible, where this is needed to define* L_{mix}. *If the null model is such that the matrix P has at least one positive entry, then the eigenvalues of* L_{mix} *are positive.*

(b) Let $L_{mix} = L_{W_{rw}} + \gamma Q_{P_{rw}}$. *If the null model is such that* $D_W = D_P$, *then the eigenvalues of* L_{mix} *are positive.*

(c) Let $L_{mix} \in \{L_{B_\gamma^+} + Q_{B_\gamma^-}, L_{B_{\gamma\,sym}^+} + Q_{B_{\gamma\,sym}^-}, L_{B_{\gamma\,rw}^+} + Q_{B_{\gamma\,rw}^-} - D_{B_\gamma^+}^{-1} D_B Q_{B_{\gamma\,rw}^-}\}$ *and assume that* $D_{B_\gamma^+}$ *and* $D_{B_\gamma^-}$ *are invertible, in those cases where this is needed to define* L_{mix}. *Assume one of the following conditions is satisfied:*

(i) the graph with adjacency matrix B_γ^+ *is connected and the matrix* B_γ^- *has at least one positive entry;*

(ii) the graph with adjacency matrix B_γ^- *is connected and the matrix* B_γ^+ *has at least one positive off-diagonal entry* $(b_\gamma^+)_{ij}$; *moreover, there exists a path along an odd number of edges from i to j in the graph with adjacency matrix* B_γ^-; *or*

(iii) the matrix B_γ^- *has positive diagonal entries.*

Then the eigenvalues of L_{mix} *are positive.*

Proof. The proof is given in Appendix B. □

Remark 5.6. The assumptions in the first two cases of Lemma 5.5 are satisfied for the NG null model.

If the graph defined by the adjacency matrix W is connected and has no self loops and P has positive diagonal elements, as is again the case for the NG null model, then, for all $i \in V$, $(b_\gamma^-)_i = \gamma p_{ii} > 0$. So (if also $D_{B_\gamma^+}$ and $D_{B_\gamma^-}$ are invertible), the conditions in the third part of the lemma are satisfied.

Remark 5.7. In general, it cannot be expected that the eigenvalues of L_{mix} are positive without additional assumptions on the graph. For example, if $L_{\text{mix}} = L_{B_\gamma^+} + Q_{B_\gamma^-}$ where

$$B = \begin{pmatrix} 0 & 1 & -1 & 0 \\ 1 & 0 & -1 & -1 \\ -1 & -1 & 0 & 1 \\ 0 & -1 & 1 & 0 \end{pmatrix},$$

it can be checked that $u \in \mathcal{V}$ with $u_1 = u_2 = 1$ and $u_3 = u_4 = -1$ is an eigenfunction (eigenvector) with eigenvalue zero.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

We now return to expression (40). When we are in case (a) of Lemma 5.1, the normalised eigenvectors ξ_i of L_{mix} are orthonormal, and thus for the coefficients in (40) we compute $c_i = \langle \xi_i, u^n \rangle$.

In case (b) of the lemma, we know by Remark 5.4 that L_{mix} has eigenvectors ξ_i that are orthonormal with respect to the inner product $\langle \cdot, \cdot \rangle_W$, and thus $c_i = \langle \xi_i, u^n \rangle_W$ instead.

Finally, in case (c) Remark 5.4 tells us that a similar conclusion holds, if we use the B_γ^+-degree-weighted inner product from (31): $c_i = \langle \xi_i, u^n \rangle_{B_\gamma^+}$.

Writing (40) fully in matrix form, we thus obtain

$$u(\tau) = \left\{ \begin{array}{ll} Xe^{-\tau\Lambda}X^T u^n, & \text{in case (a) of Lemma 5.1,} \\ Xe^{-\tau\Lambda}X^T D_W u^n = D_W^{-\frac{1}{2}} \tilde{X} e^{-\tau\Lambda} \tilde{X}^T D_W^{\frac{1}{2}} u^n, & \text{in case (b) of Lemma 5.1,} \\ Xe^{-\tau\Lambda}X^T D_{B_\gamma^+} u^n = D_{B_\gamma^+}^{-\frac{1}{2}} \tilde{X} e^{-\tau\Lambda} \tilde{X}^T D_{B_\gamma^+}^{\frac{1}{2}} u^n, & \text{in case (c) of Lemma 5.1} \end{array} \right\}$$
$$= Xe^{-\tau\Lambda}X^{-1}u^n. \tag{41}$$

We recall that the matrices X are not the same in each case, and neither are the matrices $\tilde{X}$.

5.2.2. *Implicit Euler finite-difference discretisation*

Next, we take a look at the implicit Euler finite-difference discretisation. We discretise the time domain $(0, \tau]$ into $N_t \in \mathbb{N}$ intervals $(t_{k-1}, t_k]$ ($k \in \{1, \ldots, N_t\}$) of equal length, thus $t_k = \frac{k}{N_t}\tau = k\delta t$, where $\delta t := \frac{\tau}{N_t} > 0$. Approximating u at the points t_k by $u(t_k) \approx u^k \in \mathbb{R}$ (with $u^0 = u(0)$), the discretisation of the equation $\frac{du}{dt} = -L_{\text{mix}}u$ is given by

$$\frac{u^k - u^{k-1}}{\delta t} = -L_{\text{mix}}u^k.$$

This we can rewrite as

$$u^k = (I + \delta t L_{\text{mix}})^{-1} u^{k-1}, \tag{42}$$

where I is the identity matrix of the appropriate size.

From Lemma 5.1, we recall that X is a matrix containing the normalised[23] eigenvectors of L_{mix} as columns and Λ a diagonal matrix containing the corresponding eigenvalues of L_{mix} (in the same order as the eigenvectors). By the same lemma, we know that L_{mix} is diagonalisable as $L_{\text{mix}} = X\Lambda X^{-1}$. Hence, (42) can be written as

$$u^k = \left[X(I + \delta t \Lambda)X^{-1}\right]^{-1} u^{k-1} = \left[X(I + N_t^{-1}\tau\Lambda)^{-1}X^{-1}\right]^k u^0.$$

In case (a) of Lemma 5.1, X is orthogonal and thus its inverse can be computed as $X^{-1} = X^T$. In cases (b) and (c), we have seen that $X^{-1} = \tilde{X}^T D^{\frac{1}{2}}$, for the appropriate orthogonal eigenvector matrix $\tilde{X}$ and appropriate degree matrix D.

In the linear-dynamics step of the MMBO scheme, we are interested in $u(\tau)$, for which we find

$$u(\tau) \approx u^{N_t} = \left[X(I + N_t^{-1}\tau\Lambda)^{-1}X^{-1}\right]^{N_t} u^n, \tag{43}$$

where (in a slight abuse of superscript notation) we recall that $u^0 = u(0) = u^n$, with superscript 0 indicating the initial condition for the Euler finite-difference scheme, but superscript n indicating the iteration number of the MMBO scheme. We recognise an approximation of the closed-form solution from (41) in (43).

[23] We recall that the chosen normalisation depends on the choice of L_{mix}; see Lemma 5.1.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Similarly to what we did at the end of Section 5.2.1, u^{N_t} can be written as

$$u^{N_t} = \begin{cases} \left[X \left(I + \frac{\tau}{N_t} \Lambda \right)^{-1} X^T \right]^{N_t} u^n, & \text{in case (a) of Lemma 5.1,} \\ \left[D_W^{-\frac{1}{2}} \tilde{X} \left(I + \frac{\tau}{N_t} \Lambda \right)^{-1} \tilde{X}^T D_W^{\frac{1}{2}} \right]^{N_t} u^n, & \text{in case (b) of Lemma 5.1,} \\ \left[D_{B^+}^{-\frac{1}{2}} \tilde{X} \left(I + \frac{\tau}{N_t} \Lambda \right)^{-1} \tilde{X}^T D_{B^+}^{\frac{1}{2}} \right]^{N_t} u^n, & \text{in case (c) of Lemma 5.1,} \end{cases}$$

where X and $\tilde{X}$ are the appropriate orthogonal eigenvector matrices for each case of Lemma 5.1.

5.2.3. Truncation

The occurrence of the eigenvector matrices X and eigenvalue matrices Λ in the proposed solutions in (41) and (43) allows for the use of a truncated spectrum, by which we mean that instead of $X \in \mathbb{R}^{|V| \times |V|}$ and $\Lambda \in \mathbb{R}^{|V| \times |V|}$, we will use matrices $\hat{X} \in \mathbb{R}^{|V| \times m}$ and $\Lambda \in \mathbb{R}^{m \times m}$, containing only the $m \in \mathbb{N}$ leading eigenvalues and eigenvectors.

By Lemma 5.1 we know that, for each of its possible forms, L_{mix} has real non-negative eigenvalues. When we speak of the $m \in \mathbb{N}$ leading eigenvalues, we mean the m smallest eigenvalues, counted according to multiplicity. We call the corresponding eigenvectors the m leading eigenvectors.[24]

There are several arguments for preferring a truncated method over a full method. First, smaller matrices require less storage space, which can be a significant bottleneck when dealing with very large graphs. Second, if only $m \ll |V|$ eigenvalues and eigenvectors need to be computed, this can reduce the run time of the algorithm considerably. In Section 6.4, we discuss the Nyström extension method with QR decomposition, which allows us to exploit both these benefits of truncation.

Third, in some applications it may be argued that the important information of the system under consideration is contained in the leading eigenvalues and eigenvectors, with the larger eigenvalues and corresponding eigenvectors containing more noise than signal. In that case, truncation may be viewed as a data denoising method.

Whatever the reason may be for choosing a truncated method, (40) shows us that we expect the influence of the larger eigenvalues and corresponding eigenvectors on $u(\tau)$ to be small. Unless the small error that is committed due to truncation makes the value $u(\tau)$ change sign, it will have no impact on the threshold step that follows the linear-dynamics step in the MMBO scheme.

5.3. Multiclass MMBO scheme

Just as the binary L_{mix} MMBO scheme from Section 5.1 was inspired by the Allen–Cahn-type equations from Section 4.1, so we can base a multiclass L_{mix} MMBO scheme on the multiclass Allen–Cahn-type equations from Section 4.2. We recall that $K \geq 2$ is fixed.

Multiclass L_{mix}modularity MBO scheme

- **Initialise.** Choose an initial condition $U^0 \in Pt(K)$, a 'time step' $\tau > 0$, and L_{mix} as in (38).
- **Step $n+1$: linear dynamics.** Solve the equation $\frac{dU}{dt} = -L_{\text{mix}} U$ on $(0, \tau]$ with initial condition $U(0) = U^n$.
- **Step $n+1$: threshold.** Define, for all $i \in V$, $U_{i*}^{n+1} := e^{(k^*)}$, where[25]

$$k^* \in \text{argmax}_{k \in \{1,\dots,K\}} U_{ik}(\tau).$$

[24] If the m^{th} eigenvalue has (algebraic and geometric) multiplicity strictly greater than 1, there are multiple (linearly independent) candidates for the m^{th} leading eigenvector. We anticipate that this pathological situation will not arise often in practice; if it does, an arbitrary choice among the candidate eigenvectors is made.

[25] In cases where k^* is not uniquely determined, we arbitrarily choose one of the maximisers.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

- **Stop.** Stop the scheme when a stopping condition or predetermined number of steps has been achieved.

The linear-dynamics step is a straightforward generalisation of the analogous step in the binary algorithm. The threshold step now needs to take into account that there are $K \geq 2$ clusters to choose from. We assign each node to that cluster which is represented by the highest value in the row vector $U(\tau)_{i*}$. In [20, Appendix A.3], it is proven that this procedure corresponds to first projecting the vector $U(\tau)_{i*}$ onto the $(K-1)$-simplex

$$\mathfrak{S}(K) := \left\{ w \in [-1,1]^K : \sum_{k=1}^{K} w_k = 2 - K \right\} \tag{44}$$

and then determining the nearest vertex (or vertices) of the simplex (i.e., nearest vector $e^{(k)}$) to this projected vector.

Remark 5.8. We recall from Section 3.2 that, if $K = 2$, we can relate the binary representation $u \in \mathcal{V}_{\text{bin}}$ to the multiclass representation $U \in Pt(K)$ via $U_{*1} = u$ and $U_{*2} = -u$. The outcome of the linear-dynamics step of the (binary or multiclass) MMBO scheme is in $\mathcal{V}$ or $\mathbb{R}^{|V|\times K}$, respectively, rather than in $\mathcal{V}_{\text{bin}}$ or $Pt(K)$, yet if the multiclass initial condition U^0 satisfies $U^0_{*1} = -U^0_{*2}$, then it follows that $U_{*1}(\tau) = -U_{*1}(\tau)$, since both vectors are solutions to the same system of linear ordinary differential equations (ODEs). Because u is a solution to the same system of ODEs, we can still make the identification $U_{*1} = u = -U_{*2}$. This means that, for all $i \in V$, $U_{i1}(\tau) \geq 0$ if and only if $U_{i2}(\tau) \leq 0$, which in turn is equivalent to $u_i(\tau) \geq 0$. Thus, $U_{i1}(\tau) \geq U_{i2}(\tau)$ if and only if $u_i(\tau) \geq 0$, which makes the binary and multiclass threshold steps equivalent (up to non-uniqueness issues when $U_{i1}(\tau) = U_{i2}(\tau)$).

5.4. Numerical schemes for multiclass MMBO

We briefly discuss the generalisations of the numerical schemes that we encountered in Section 5.2 to the multiclass case. An in-depth look at the resulting algorithms follows in Section 6.

5.4.1. Closed-form matrix exponential solution

If the relevant (for the case at hand) assumptions from Lemma 5.1 are satisfied, then the closed-form solution of the multiclass-linear-dynamics step is a straightforward generalisation of the solution in the binary case, since each column of U satisfies the same linear system of ODEs and there is no coupling between the dynamics of different columns. Thus, $U(\tau) = e^{-\tau L_{\text{mix}}} U^n$ and from there the same arguments as in Section 5.2.1 lead to

$$U(\tau) = X e^{-\tau \Lambda} X^{-1} U^n, \tag{45}$$

where we recall that the meaning of X depends on which of the three cases of Lemma 5.1 L_{mix} satisfies.

5.4.2. Implicit Euler finite-difference discretisation

For the same reasons as mentioned in Section 5.4.1, the Euler finite-difference scheme from Section 5.2.2 also straightforwardly generalises from the binary to the multiclass setting, assuming that the relevant assumptions of Lemma 5.1 are satisfied. Hence, we find, from (43), that

$$U(\tau) \approx \left[X (I + N_t^{-1} \tau \Lambda)^{-1} X^{-1} \right]^{N_t} U^n, \tag{46}$$

where we recall that $N_t \in \mathbb{N}$ is the number of steps in the finite-difference discretisation of the interval $[0, \tau]$.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

6. The modularity MBO algorithms

In Section 5, we established multiclass MMBO schemes for clustering into $K \geq 2$ clusters and explored numerical methods for the computation of such schemes. Based on the work done in Section 5, in this section we present the algorithms we use in detail. The results of applying these algorithms to various data sets are presented in Section 7.

We present the algorithm based on the closed-form solution of Section 5.4.1 in Section 6.2 and the algorithm based on the Euler finite-difference scheme of Section 5.4.2 in Section 6.3.

From Section 5.2.3, we recall that it can be beneficial to use only the m leading eigenvalues and eigenvectors of L_{mix}, rather than its full spectrum. In Section 6.4, we present the Nyström extension method with QR decomposition, which provides an efficient way to approximate these leading eigenvalues and eigenvectors.

As usual we assume that L_{mix} is as in (38) and that the corresponding assumptions from Lemma 5.1 are satisfied. Additionally, we assume that L_{mix} has positive eigenvalues, which in particular guarantees that the smallest eigenvalue of L_{mix} is positive. Lemma 5.5 gives sufficient conditions for this assumption to be satisfied.

For notational convenience, we write

$$(F, H) \in \left\{(W, P), (B_\gamma^+, B_\gamma^-)\right\}. \tag{47}$$

The choice of L_{mix} determines the choice of the pair (F, H) to be what is needed for the construction of L_{mix}.

Besides L_{mix} and the corresponding choice of matrices F and H, the parameters that are required as input for our algorithms are the following:

- The maximum number of non-empty clusters $K \geq 2$.
- The resolution parameter $\gamma > 0$ that determines, via (15), which modularity function we are attempting to maximise. It also determines the time step τ in the MMBO scheme, via the method from [7], which we explain in more detail in Section 6.1 below.
- The number $m \in \{1, \ldots, |V|\}$ of leading eigenvalues and corresponding eigenvectors that we use.
- $\eta \in (0, \infty)$ which determines the stopping criterion; we choose either a *partition-based stopping criterion*, under which the algorithm terminates if

$$\frac{\max\limits_{i \in V} ||U_{i*}^{n+1} - U_{i*}^n||_2^2}{\max\limits_{i \in V} ||U_{i*}^{n+1}||_2^2} < \eta, \tag{48}$$

  as in [24], or a *modularity-based stopping criterion* under which the iteration terminates if

$$\left|\mathcal{Q}_{\text{mul},\gamma}(U^{n+1}; W, P) - \mathcal{Q}_{\text{mul},\gamma}(U^n; W, P)\right| < \eta. \tag{49}$$

- Only for Algorithm 2 that uses the Euler finite-difference scheme for the linear-dynamics step: $N_t \in \mathbb{N}$ is the number of steps in the finite-difference discretisation of $[0, \tau]$.

We recall that, given a choice of L_{mix}, Λ is the diagonal matrix containing the eigenvalues of L_{mix} on its diagonal and X is a matrix containing corresponding eigenvectors of L_{mix} as columns in the order corresponding to that of the eigenvalues in Λ. The required normalisation of these eigenvectors differs as presented in cases (a)–(c) of Lemma 5.1. In cases (b) and (c), Lemma 5.1 provides expressions that allow computation of X in terms of a matrix $\tilde{X}$ with Euclidean-normalised columns.

6.1. Choice of the time step

To avoid having to manually fine-tune the value of the time step τ in our algorithms, we follow the method proposed by Boyd *et al.* [7, Section 4.5].

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Two main considerations are of importance in the choice of τ. If the value is chosen too small, then the linear-dynamics step of the MBO scheme will change the initial condition very little and the threshold step will return the initial condition again. In other words, the initial condition will be stationary under the MBO scheme. If the value of τ is chosen too large, then (because all eigenvalues of L_{mix} are positive) the result of one application of the linear-dynamics step will be a function which is close to zero on all nodes of the graph. Only the structure contained in the mode(s) with the smallest eigenvalue(s) is retained, which is typically not sufficient to find optimal communities. Moreover, to have a threshold step which is robust to noise, we prefer the values on the nodes to be clearly separated and not all clustered together near zero. Thus, we need to choose a value of τ which is neither too small nor too large. The details of what is 'too small' and 'too large' depend on the structure of the graph and the choice of initial condition. In [7], a specific choice of τ is suggested for the particular MBO-type scheme that is employed in that paper. Adapting a method from Van Gennip *et al.* [73], upper (τ_{upp}) and lower (τ_{low}) bounds are established[26] by [7] and their geometric mean $\sqrt{\tau_{\text{low}}\tau_{\text{upp}}}$ is used as the value of τ. For the lower bound, τ_{low} is computed for $K = 2$ (even in cases where $K \neq 2$). An explicit numerical value for general K is harder to obtain and [7] expects (without proof) that the case $K = 2$ presents the worst-case scenario. In the next lemma, we adapt the method from [7] to our MBO scheme and give upper and lower bounds on τ (which are not expected to be sharp but are efficiently computable).

Lemma 6.1. *Let $U^0 \in Pt(K)$, $\tau > 0$, and let L_{mix} be as in (38). Assume U solves $\frac{dU}{dt} = -L_{mix}U$ on $(0, \tau]$ with initial condition $U(0) = U^0$. Write U^1 for the outcome after one iteration of the multiclass L_{mix} modularity MBO scheme starting from U^0.*[27]

(a) Then

$$\|U(\tau) - U^0\|_\infty \leq K\left(e^{\tau\|L_{mix}\|_\infty} - 1\right).$$

We have the following (non-sharp[28]*) upper bounds on $\|L_{mix}\|_\infty$:*

(i) If $L_{mix} = L_W + \gamma Q_P$, then $\|L_{mix}\|_\infty \leq L^{max} := 2(d_{W,max} + \gamma d_{P,max})$.

(ii) If $L_{mix} = L_{W_{sym}} + \gamma Q_{P_{sym}}$, then $\|L_{mix}\|_\infty \leq L^{max} := 1 + \gamma + (d_{W,min})^{-\frac{1}{2}}(d_{W,max})^{\frac{1}{2}} + \gamma(d_{P,min})^{-\frac{1}{2}}(d_{P,max})^{\frac{1}{2}}$.

(iii) If $L_{W_{rw}} + \gamma Q_{P_{rw}}$, then $\|L_{mix}\|_\infty \leq L^{max} := 2(1 + \gamma)$.

(iv) If $L_{B_\gamma^+} + Q_{B_\gamma^-}$, then $\|L_{mix}\|_\infty \leq L^{max} := 2(d_{B_\gamma^+,max} + d_{B_\gamma^-,max})$.

(v) If $L_{mix} = L_{B_{\gamma\,sym}^+} + Q_{B_{\gamma\,sym}^-}$, then $\|L_{mix}\|_\infty \leq L^{max} := 2 + (d_{B_\gamma^+,min})^{-\frac{1}{2}}(d_{B_\gamma^+,max})^{\frac{1}{2}} + (d_{B_\gamma^-,min})^{-\frac{1}{2}}(d_{B_\gamma^-,max})^{\frac{1}{2}}$.

(vi) If $L_{mix} = L_{B_{\gamma\,rw}^+} + Q_{B_{\gamma\,rw}^-} - D_{B_\gamma^+}^{-1} D_{B_\gamma} Q_{B_{\gamma\,rw}^-}$, then $\|L_{mix}\|_\infty \leq L^{max} := 2(1 + d_{B_\gamma^+,min}^{-1} d_{B_\gamma^-,max})$.

(b) If $K = 2$ and

$$\tau < \tau_{low} := (L^{max})^{-1} \ln(2),$$

then $U^1 = U^0$.

(c) Assume that the assumptions of Lemma 5.1 are satisfied. If λ_1 is the minimal eigenvalue of L_{mix}, then

$$\|U(\tau)\|_{Fr,C} \leq e^{-\tau\lambda_1}\|U^0\|_{Fr},$$

where the matrix $C \in \mathbb{R}^{|V|\times|V|}$ satisfies:

(i) if $L_{mix} \in \{L_W + \gamma Q_P, L_{W_{sym}} + \gamma Q_{P_{sym}}, L_{B_\gamma^+} + Q_{B_\gamma^-}, L_{B_{\gamma\,sym}^+} + Q_{B_{\gamma\,sym}^-}\}$, then $C = I$;

(ii) if $L_{mix} = L_{W_{rw}} + \gamma Q_{P_{rw}}$, then $C = D_W$; and

(iii) if $L_{mix} = L_{B_{\gamma\,rw}^+} + Q_{B_{\gamma\,rw}^-} - D_{B_\gamma^+}^{-1} D_{B_\gamma} Q_{B_{\gamma\,rw}^-}$, then $C = D_{B_\gamma^+}$.

[26] These bounds are not believed to be sharp.

[27] That is, after 'step 1: linear dynamics' and 'step 1: threshold'.

[28] The bounds can be improved by using an estimate earlier in the chain of estimates that appear in the proof.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Algorithm 1. The MMBO scheme using the closed-form solution of the linear-dynamics step

Require: $K \geq 2$, $m \in \{1, \ldots, |V|\}$, $\gamma > 0$, $F, H \in [0, \infty)^{|V| \times |V|}$ as in (47),
stopping criterion (48) or (49) with $\eta > 0$

Initialise:

$D_F, D_H \leftarrow F, H$ ▷ Compute D_F and D_H (and, if needed, D_F^{-1}, $D_F^{-\frac{1}{2}}$, D_H^{-1}, and $D_H^{-\frac{1}{2}}$)

$L_{\text{mix}} \leftarrow F, H\,(D_F, D_H, \gamma)$ ▷ Compute L_{mix} from (38); some cases do not need D_F, D_H, γ

$\Lambda, X \leftarrow L_{\text{mix}}, m$ ▷ Eigendecomposition of L_{mix}; columns of X normalised as in Lemma 5.1; use only leading m eigenvalues and eigenvectors

$\lambda_1 \leftarrow \Lambda$ ▷ Select the minimal eigenvalue

$U^0 = (-1)_{|V| \times K}$

for $i = 1 \rightarrow K$ **do**

$u^0_{iK} = 1$ ▷ Randomly select K nodes and assign each one to a different cluster

end for

for $j = 1 \rightarrow (|V| - K)$ **do**

$l = random.sample(K, 1)$

$u^0_{jl} = 1$ ▷ Assign each node randomly to a community in initial condition

end for

$d_{\max} \leftarrow D$ ▷ Select the maximal degree of D

$\tau_{\text{low}} \leftarrow (L^{\max})^{-1} \ln(2)$ ▷ Compute τ_{low} as in Lemma 6.1 part (b)

$\tau_{\text{upp}} \leftarrow \lambda_1^{-1} \ln\left(K^{\frac{1}{2}} c_{\min}^{-\frac{1}{2}} \theta^{-1} \|U^0\|_{\text{Fr}}\right)$ ▷ Compute τ_{low} as in Lemma 6.1 part (d)

$\tau \leftarrow \sqrt{\tau_{\text{low}} \tau_{\text{upp}}}$

$M_{exp} \leftarrow \exp(-\tau \Lambda)$

$n \leftarrow 0$

while Stopping criterion not satisfied and $n \leq 10000$ **do**

Linear dynamics:

$U^{n+\frac{1}{2}} \leftarrow X M_{exp} X^{-1} U^n$ ▷ Compute (45)

Thresholding:

$(U_i)^{n+1} = \underline{e}_j$ with $j \in \underset{l=1,\ldots,K}{\text{argmax}} \left\{(u_{il})^{n+\frac{1}{2}}\right\}$

$n = n + 1$

end while

(d) Assume the assumptions of Lemma 5.1 to be satisfied and $\lambda_1 \neq 0$. Let $\theta > 0$ and define $c_{min} := \min_{i \in V} C_{ii}$ with C as in part (c) of this lemma. If

$$\tau > \tau_{upp} := \lambda_1^{-1} \ln\left(K^{\frac{1}{2}} c_{min}^{-\frac{1}{2}} \theta^{-1} \|U^0\|_{Fr}\right),$$

then $\|U(\tau)\|_\infty < \theta$.

Proof. We give the proof in Appendix C. □

We follow [7] in choosing $\tau = \sqrt{\tau_{\text{low}} \tau_{\text{upp}}}$.

6.2. MMBO scheme using the closed-form solution of the linear-dynamics step

We recall from Section 5.4.1 that, to employ the closed-form solution of the linear-dynamics step to compute the multiclass L_{mix} MMBO scheme, we need to compute the expression in (45). The resulting algorithm is summarised in Algorithm 1. The command $random.sample(K, N)$ returns a list of length N, with uniformly random sampling from 1 to K of N distinct elements ($N \leq K$).

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Algorithm 2. The MMBO scheme using the Euler finite-difference discretisation

Require: $K \geq 2$, $m \in \{1, \ldots, |V|\}$, $\gamma > 0$, $F, H \in [0, \infty)^{|V|\times|V|}$ as in (47), $N_t \in \mathbb{N}$, stopping criterion (48) or (49) with $\eta > 0$

Initialise:

$D_F, D_H \leftarrow F, H$ ▷ Compute D_F and D_H (and, if needed, D_F^{-1}, $D_F^{-\frac{1}{2}}$, D_H^{-1}, and $D_H^{-\frac{1}{2}}$)

$L_{\text{mix}} \leftarrow F, H\,(D_F, D_H, \gamma)$ ▷ Compute L_{mix} from (38); some cases do not need D_F, D_H, γ

$\Lambda, X \leftarrow L_{\text{mix}}, m$ ▷ Eigendecomposition of L_{mix}; columns of X normalised as in Lemma 5.1; use only leading m eigenvalues and eigenvectors

$\lambda_1 \leftarrow \Lambda$ ▷ Select the minimal eigenvalue

$U^0 = (-1)_{|V|\times K}$

for $i = 1 \rightarrow K$ **do**

$u_{iK}^0 = 1$ ▷ Randomly select K nodes and assign each one to a different cluster

end for

for $i = 1 \rightarrow (|V| - K)$ **do**

$l = random.sample(K, 1)$

$u_{il}^0 = 1$ ▷ Assign each node randomly to a community in initial condition

end for

$d_{\max} \leftarrow D$ ▷ Select the maximal degree of D

$\tau_{\text{low}} \leftarrow (L^{\max})^{-1} \ln(2)$ ▷ Compute τ_{low} as in Lemma 6.1 part (b)

$\tau_{\text{upp}} \leftarrow \lambda_1^{-1} \ln\left(K^{\frac{1}{2}} c_{\min}^{-\frac{1}{2}} \theta^{-1} \|U^0\|_{\text{Fr}}\right)$ ▷ Compute τ_{low} as in Lemma 6.1 part (d)

$\tau \leftarrow \sqrt{\tau_{\text{low}} \tau_{\text{upp}}}$

$n \leftarrow 0$

while Stopping criterion not satisfied and $n \leq 10000$ **do**

Linear dynamics:

for $s = 1 \rightarrow N_t$ **do**

$U^{n+\frac{1}{2}} \leftarrow \left[X\left(I + \frac{\tau}{N_t}\Lambda\right)^{-1} X^{-1}\right]^{N_t} U^n$ ▷ Compute diffusion via (46)

$s = s + 1$

end for

Thresholding:

$(U_i)^{n+1} = \underline{e}_j$ with $j \in \underset{l=1,\ldots,K}{\operatorname{argmax}} \left\{(u_{il})^{n+\frac{1}{2}}\right\}$

$n = n + 1$

end while

6.3. Alternative variant of the MMBO scheme

From Section 5.4.2, we recall that for the MMBO scheme with the Euler finite-difference scheme, we have to solve equation (46).

We use the same thresholding step and stopping conditions for this alternative variant as we do for the MMBO scheme in Algorithm 1. The MMBO scheme using the Euler finite-difference discretisation is summarised in Algorithm 2. The main difference between Algorithm 1 and Algorithm 2 is the diffusion step, as seen in (45) and (46).

6.4. Nyström extension with QR decomposition

Both Algorithm 1 and Algorithm 2 contain steps that require us to compute the m leading eigenvalues and corresponding eigenvectors of L_{mix}. In the examples we consider in Section 7, the sizes of the graphs go up to tens of thousands of nodes, making it time-consuming to perform operations on the matrix.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

The Nyström approximation [58], which generates a low-rank approximation of the original matrix from a subset of its columns, is an effective way to tackle this issue. The choice of sampling method could affect the Nyström approximation performance, since different samples provide different approximations of the original adjacency matrix W.

Before applying the Nyström method to our particular choices for the matrix L_{mix}, we first give a brief explanation of the method for a general, symmetric matrix $C \in \mathbb{R}^{|V|\times|V|}$.

We sample k distinct points[29] uniformly at random from $|V|$ points and partition the matrix C as

$$C = \begin{pmatrix} C_{11} & C_{21}^T \\ C_{21} & C_{22} \end{pmatrix}, \tag{50}$$

where $C_{11} \in \mathbb{R}^{k\times k}$, $C_{21} \in \mathbb{R}^{(|V|-k)\times k}$, and $C_{22} \in \mathbb{R}^{(|V|-k)\times(|V|-k)}$. We have relabelled the points such that our k sampled points are the first k points. We note that C_{11} and C_{22} are symmetric.

Because C_{11} is a real and symmetric matrix, we can perform an eigenvalue decomposition to obtain $C_{11} = U\Lambda_k U^T$, where[30] $\Lambda_k := \operatorname{diag}(\lambda_1, \ldots, \lambda_k) \in \mathbb{R}^{k\times k}$ is a diagonal matrix with the k eigenvalues (counted according to multiplicity) λ_i of C_{11} on its diagonal and $U \in \mathbb{R}^{k\times k}$ is an orthogonal matrix which has the corresponding eigenvectors of C_{11} as columns (in the order corresponding to the order of the eigenvalues in Λ_k).

We write $C_{11}^\dagger$ for the Moore–Penrose pseudoinverse[31] of C_{11}. If C_{11} is invertible, then the pseudoinverse is equal to the inverse of C_{11}. If C_{11} is not invertible, then the relationships $C_{11}^\dagger C_{11} C_{11}^\dagger = C_{11}^\dagger$ and $C_{11} C_{11}^\dagger C_{11} = C_{11}$ still hold. The pseudoinverse can be computed as $C_{11}^\dagger = U\Lambda_k^\dagger U^T$, where $\Lambda_k^\dagger$ is the pseudoinverse. Specifically, this means that $\Lambda_k^\dagger$ is a diagonal matrix whose diagonal elements are the reciprocals of the diagonal elements of Λ_k when those are non-zero, and zero when those are zero.

We would like to use the columns of U to approximate k eigenvectors of C. Eigenvectors of C are in $\mathbb{R}^{|V|}$, whereas the columns of U are in $\mathbb{R}^k$; the main idea of the Nyström extension is to approximate the 'missing' $|V| - k$ entries by $C_{21}U\Lambda_k^\dagger$. This approximation is inspired by a quadrature rule; for more details, we refer to Fowlkes *et al.* [23] and Bertozzi and Flenner [4]. Thus,

$$U_C := \begin{pmatrix} U \\ C_{21}U\Lambda_k^\dagger \end{pmatrix} \in \mathbb{R}^{|V|\times k}$$

is an approximation of k eigenvectors of C, which in turn gives us an approximation of the full matrix C:

$$\begin{aligned} \bar{C} := U_C \Lambda_k U_C^T &= \begin{pmatrix} U \\ C_{21}U\Lambda_k^\dagger \end{pmatrix} \Lambda_k \begin{pmatrix} U \\ C_{21}U\Lambda_k^\dagger \end{pmatrix}^T \\ &= \begin{pmatrix} U\Lambda_k U^T & U\Lambda_k\Lambda_k^\dagger U^T C_{21}^T \\ C_{21}U\Lambda_k^\dagger\Lambda_K U^T & C_{21}U\Lambda_k^\dagger\Lambda_k\Lambda_k^\dagger U^T C_{21}^T \end{pmatrix} = \begin{pmatrix} C_{11} & C_{11}C_{11}^\dagger C_{21}^T \\ C_{21}C_{11}^\dagger C_{11} & C_{21}C_{11}^\dagger C_{21}^T \end{pmatrix} \\ &= \begin{pmatrix} C_{11} \\ C_{21} \end{pmatrix} C_{11}^\dagger \begin{pmatrix} C_{11} & C_{21}^T \end{pmatrix}. \end{aligned} \tag{51}$$

Comparing (50) and (51), we obtain that C_{22} is approximated as

$$C_{22} \approx C_{21}C_{11}^\dagger C_{21}^T.$$

[29] We emphasise that, despite what the notation might suggest, in the context of this section about the Nyström extension, k does not need to be related to clusters.

[30] If $v \in \mathbb{R}^k$ is a vector, $\operatorname{diag}(v) \in \mathbb{R}^{k\times k}$ is a diagonal matrix with diagonal entries $(\operatorname{diag}(v))_{ii} = v_i$.

[31] Which can be computed, for example, by using the singular value decomposition of C_{11}.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

We note that in many sources, the off-diagonal blocks are often given as C_{21} and C_{21}^T, rather than $C_{21}C_{11}^\dagger C_{11}$ and $C_{11}C_{11}^\dagger C_{21}^T$, respectively. If C_{11} is invertible, these are equivalent, but in general they need not be.[32] In our numerical tests in Section 7, the matrices that are used for C_{11} are always invertible.

Now we wish to find the eigenvalue decomposition of $\bar{C}$.[33] We follow the QR decomposition method introduced in Budd *et al.* [12], which uses the thin QR decomposition [27, Theorems 5.2.2 and 5.2.3]:

$$\begin{pmatrix} C_{11} \\ C_{21} \end{pmatrix} = \tilde{Q}R,$$

where[34] $\tilde{Q} \in \mathbb{R}^{|V|\times k}$ has orthonormal columns (i.e., $\tilde{Q}^T\tilde{Q} = I$) and $R \in \mathbb{R}^{k\times k}$ is an upper triangular matrix with positive diagonal entries. This decomposition is possible if $\begin{pmatrix} C_{11} \\ C_{21} \end{pmatrix}$ has full column rank, that is, if its column rank is k. We note that this is guaranteed to be the case[35] if C_{11} is invertible, and thus in particular in our numerical studies in Section 7.

By (51), we see that $\bar{C} = QRC_{11}^\dagger(QR)^T$. The matrix $RC_{11}^\dagger R^T$ is real and symmetric and thus admits an eigendecomposition:

$$RC_{11}^\dagger R^T = \Upsilon\Sigma\Upsilon^T,$$

with $\Upsilon \in \mathbb{R}^{k\times k}$ orthogonal and $\Sigma \in \mathbb{R}^{k\times k}$ diagonal. Then

$$\bar{C} = \tilde{Q}RC_{11}^\dagger R^T\tilde{Q}^T = \tilde{Q}\Upsilon\Sigma\Upsilon^T\tilde{Q}^T = \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T. \tag{52}$$

Since $(\tilde{Q}\Upsilon)^T\tilde{Q}\Upsilon = I$, we have that $\tilde{Q}\Upsilon \in \mathbb{R}^{|V|\times k}$ has orthonormal columns. Thus, we can view the k diagonal entries of Σ as approximate eigenvalues of $\bar{C}$ (and thus of C) with the columns of $\tilde{Q}\Upsilon$ the corresponding approximate eigenvectors.

Remark 6.2. The method explained above can be used to estimate k eigenvalues and corresponding eigenvectors of the matrix C based on sampling the submatrices C_{11} and C_{21}. In our applications, we require eigenvalues and eigenvectors of the matrix $D_C + C$ or of a normalised matrix $D_C^{-\frac{1}{2}}CD^{-\frac{1}{2}}$ or $D_C^{-1}C$, where $D_C = \mathrm{diag}(d_C)$ is the diagonal degree matrix based on C with diagonal entries $(d_C)_i$ as in (1). Because we cannot compute D_C exactly if we do not have access to all entries of C, we first have to compute an approximation to D_C. We set $\bar{D}_C := \mathrm{diag}(\bar{d}_C)$ with

$$\bar{d}_C := \bar{C}\mathbf{1}_{|V|} = \bar{C}\begin{pmatrix} \mathbf{1}_k \\ \mathbf{1}_{|V|-k} \end{pmatrix} = \begin{pmatrix} C_{11}\mathbf{1}_k + C_{11}C_{11}^\dagger C_{21}^T\mathbf{1}_{|V|-k} \\ C_{21}C_{11}^\dagger C_{11}\mathbf{1}_k + C_{21}C_{11}^\dagger C_{21}^T\mathbf{1}_{|V|-k} \end{pmatrix},$$

where $\mathbf{1}_k \in \mathbb{R}^k$ is the k-dimensional column vector whose entries are all $\mathbf{1}$ (see [24]).[36]

[32] The operator $I - C_{11}^\dagger C_{11}$ is the projection operator onto the kernel of C_{11}. Since $C_{21}C_{11}^\dagger C_{11} = C_{21} - C_{21}(I - C_{11}^\dagger C_{11})$, if the kernel of C_{11} is a subset of the kernel of C_{21}, then $C_{21}C_{11}^\dagger C_{11} = C_{21}$. Under the same assumption, we also have $C_{11}C_{11}^\dagger C_{21}^T = C_{21}^T - (I - C_{11}C_{11}^\dagger)C_{21}^T = \left[C_{21}(I - C_{11}^\dagger C_{11})\right]^T = C_{21}^T$.

[33] By this we mean that we wish to find a matrix $O \in \mathbb{R}^{|V|\times k}$ with orthonormal columns and a diagonal matrix $\Sigma \in \mathbb{R}^{k\times k}$ such that $\bar{C} = O\Sigma O^T$. By construction in (51), the rank of $\bar{C}$ is at most k, hence a full eigendecomposition can be obtained from O and Σ by extending O to a $|V|$-by-$|V|$ orthogonal matrix (e.g., via the Gram–Schmidt process) and padding Σ with zeroes to form a diagonal $|V|$-by-$|V|$ matrix. We note that $\bar{C} = U_C\Lambda_k U_C^\dagger$ does not give such an eigenvalue decomposition, as U_C does typically not have orthonormal columns: $U_C^T U_C = I + \Lambda_k^\dagger U^T C_{21}^T C_{21} U\Lambda_k^\dagger$.

[34] We write $\tilde{Q}$ instead of Q to distinguish this matrix from the various other Qs that are used in this paper.

[35] If the column rank is not full, in theory one could remove columns from $\begin{pmatrix} C_{11} \\ C_{21} \end{pmatrix}$ until it does have full column rank. In practice, however, if one has to remove many columns to achieve this, a resampling of the k columns might be a better idea, if possible.

[36] Recalling footnote 32, we find that if $\mathbf{1}_k$ is orthogonal to the kernel of C_{11} (or equivalently, if $\mathbf{1}_k$ is in the column space of C_{11}), then $C_{21}C_{11}^\dagger C_{11}\mathbf{1}_k = C_{21}\mathbf{1}_k$. Similarly, since $I - C_{11}C_{11}^\dagger$ is the projection operator onto the kernel of $C_{11}^T = C_{11}$, if $C_{21}^T\mathbf{1}_{|V|-k}$ is orthogonal to the kernel of C_{11}, then $C_{11}C_{11}^\dagger C_{21}^T\mathbf{1}_{|V|-k} = C_{21}^T\mathbf{1}_{|V|-k}$.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Now we approximate $D_C + C$ by $\bar{D}_C + \bar{C}$ and $D_C^{-1}C$ by $\bar{D}_C^{\dagger}\bar{C}$. If $\bar{D}_C$ has non-zero diagonal elements, then $\bar{D}_C^{\dagger} = \bar{D}_C^{-1}$. Moreover, we approximate $D_C^{-\frac{1}{2}} C D^{-\frac{1}{2}}$ by $(\bar{D}_C^{\dagger})^{\frac{1}{2}} \bar{C} (\bar{D}_C^{\dagger})^{\frac{1}{2}}$, where we require $\bar{D}_C^{\dagger}$ (and thus also $\bar{D}_C$) to have non-negative diagonal elements, for the square root to be well defined.

In general, there is no guarantee that $\bar{D}_C$ has non-zero or non-negative diagonal elements, but all matrices of type $\bar{D}_C$ that we use in our numerical tests in Section 7 do. The following observation may be of use establishing these properties in specific situations. If we denote by $d_{C_{1:k}} := C_{11}\mathbf{1}_k + C_{21}^T \mathbf{1}_{|V|-k} \in \mathbb{R}^k$, the column vector containing the first k entries of the actual degree vector $d_C \in \mathbb{R}^{|V|}$, then

$$\bar{d}_C = \begin{pmatrix} C_{11}C_{11}^{\dagger} d_{C_{1:k}} \\ C_{21}C_{11}^{\dagger} d_{C_{1:k}} \end{pmatrix}.$$

If C is the adjacency matrix of a graph with positive degrees for all nodes (e.g., a connected graph), then all entries of $\bar{d}_C$ will be positive if the matrices $C_{11}C_{11}^{\dagger}$ and $C_{21}C_{11}^{\dagger}$ both preserve entrywise positivity of vectors. If C_{11} is invertible, this is clearly the case for $C_{11}C_{11}^{\dagger}$. In fact, in that case the first k entries of $\bar{d}_C$ are exactly equal to the first k entries of the actual degree vector d_C.

To obtain approximate eigenvalues and eigenvectors of $D_C + C$, $D_C^{-\frac{1}{2}} C D^{-\frac{1}{2}}$, or $D_C^{-1}C$, we can now apply the decomposition from (52) to $D_C + C$, $(\bar{D}_C^{\dagger})^{\frac{1}{2}} \bar{C} (\bar{D}_C^{\dagger})^{\frac{1}{2}}$ or $\bar{D}_C^{\dagger}\bar{C}$, respectively, instead of to $\bar{C}$.

Now we apply the decomposition in (52) (using the approximate normalisations from Remark 6.2 where needed) to the six different cases for the matrix L_{mix} that we will encounter. In each of the cases below, the matrices $\tilde{Q}$, Υ and Σ are different, corresponding via (52) to the specific matrix $\bar{C}$ for which the decomposition is computed.

- $\mathbf{L}_{\text{mix}} = \mathbf{L}_{\mathbf{W}} + \gamma \mathbf{Q}_{\mathbf{P}}$. We use (51) to find approximations $\bar{W}$ and $\bar{P}$ of W and P, respectively, and approximate degree matrices $\bar{D}_W$ and $\bar{D}_P$ as in Remark 6.2. Then we apply (52) to $\bar{D}_W + \bar{W} + \gamma(\bar{D}_P + \bar{P})$ to find

$$L_{\text{mix}} \approx \bar{D}_W + \bar{W} + \gamma(\bar{D}_P + \bar{P}) = \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T.$$

  Thus, Σ has the approximate eigenvalues on its diagonal and the approximate eigenvectors are the columns of $\tilde{Q}\Upsilon$.

- $\mathbf{L}_{\text{mix}} = \mathbf{L}_{\mathbf{B}_\gamma^+} + \mathbf{Q}_{\mathbf{B}_\gamma^-}$. We proceed as in the previous case, with B_γ^+ instead of W and B_γ^- instead of γP. Then

$$L_{\text{mix}} \approx \bar{D}_{B_\gamma^+} + \bar{B}_\gamma^+ + \bar{D}_{B_\gamma^-} + \bar{B}_\gamma^- = \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T.$$

  The diagonal of Σ gives the approximate eigenvalues with the approximate eigenvectors being the columns of $\tilde{Q}\Upsilon$.

- $\mathbf{L}_{\text{mix}} = \mathbf{L}_{\mathbf{W}_{\text{sym}}} + \gamma \mathbf{Q}_{\mathbf{P}_{\text{sym}}}$. Again we use (51) to find approximations $\bar{W}$ and $\bar{P}$ of W and P, respectively, and approximate $\bar{D}_W$ and $\bar{D}_P$ as in Remark 6.2. Then we approximate $D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}} - \gamma D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}}$ by $(\bar{D}_W^{\dagger})^{\frac{1}{2}} \bar{W} (\bar{D}_W^{\dagger})^{\frac{1}{2}} - \gamma (\bar{D}_P^{\dagger})^{\frac{1}{2}} \bar{P} (\bar{D}_P^{\dagger})^{\frac{1}{2}}$ and use (52) to obtain

$$\begin{aligned} L_{\text{mix}} &= (1+\gamma)I - \left(D_W^{-\frac{1}{2}} W D_W^{-\frac{1}{2}} - \gamma D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}}\right) \\ &\approx (1+\gamma)I - \left((\bar{D}_W^{\dagger})^{\frac{1}{2}} \bar{W} (\bar{D}_W^{\dagger})^{\frac{1}{2}} - \gamma (\bar{D}_P^{\dagger})^{\frac{1}{2}} \bar{P} (\bar{D}_P^{\dagger})^{\frac{1}{2}}\right) = (1+\gamma)I - \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T \\ &= \tilde{Q}\Upsilon\big[(1+\gamma)I - \Sigma\big](\tilde{Q}\Upsilon)^T. \end{aligned}$$

  We use the diagonal elements of $(1+\gamma)I - \Sigma$ as approximate eigenvalues of L_{mix} and the columns of $\tilde{Q}\Upsilon$ as corresponding approximate eigenvectors.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

- $\mathbf{L}_{\text{mix}} = \mathbf{L}_{\mathbf{B}^+_{\gamma\,\text{sym}}} + \mathbf{Q}_{\mathbf{B}^-_{\gamma\,\text{sym}}}$. In this case, we proceed as in the previous one, with B^+_γ instead of W and B^-_γ instead of γP. Then

$$\begin{aligned} L_{\text{mix}} &= 2I - \left(D_{B^+_\gamma}^{-\frac{1}{2}} B^+_\gamma D_{B^+_\gamma}^{-\frac{1}{2}} - D_{B^-_\gamma}^{-\frac{1}{2}} B^-_\gamma D_{B^-_\gamma}^{-\frac{1}{2}}\right) \\ &\approx 2I - \left((\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}} \bar{B}^+_\gamma (\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}} - (\bar{D}^\dagger_{B^-_\gamma})^{\frac{1}{2}} \bar{B}^-_\gamma (\bar{D}^\dagger_{B^-_\gamma})^{\frac{1}{2}}\right) = 2I - \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T \\ &= \tilde{Q}\Upsilon\big[2I - \Sigma\big](\tilde{Q}\Upsilon)^T. \end{aligned}$$

  Hence, we obtain the approximate eigenvalues from $2I - \Sigma$ with the columns of $\tilde{Q}\Upsilon$ being the approximate eigenvectors.

- $\mathbf{L}_{\text{mix}} = \mathbf{L}_{\mathbf{W}_{\text{rw}}} + \gamma \mathbf{Q}_{\mathbf{P}_{\text{rw}}}$. We compute approximations $\bar{W}$ and $\bar{P}$ of W and P, respectively, via (51) and approximate $\bar{D}_W$ and $\bar{D}_P$ as in Remark 6.2. Then we compute the decomposition from (52) for $\bar{D}_W^\dagger \bar{W} - \gamma \bar{D}_P^\dagger \bar{P}$ to find

$$\begin{aligned} L_{\text{mix}} &= (1+\gamma)I - D_W^{-1}W + D_P^{-1}P \approx (1+\gamma)I - \left(\bar{D}_W^\dagger \bar{W} - \gamma \bar{D}_P^\dagger \bar{P}\right) \\ &= (1+\gamma)I - \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T = \tilde{Q}\Upsilon\big[(1+\gamma)I - \Sigma\big](\tilde{Q}\Upsilon)^T. \end{aligned}$$

  We get the approximate eigenvalues from $(1+\gamma)I - \Sigma$ with the columns of $\tilde{Q}\Upsilon$ as approximate eigenvectors.

- $\mathbf{L}_{\text{mix}} = \mathbf{L}_{\mathbf{B}^+_{\gamma\,\text{rw}}} + \mathbf{Q}_{\mathbf{B}^-_{\gamma\,\text{rw}}} - \mathbf{D}^{-1}_{\mathbf{B}^+_\gamma} \mathbf{D}_{\mathbf{B}_\gamma} \mathbf{Q}_{\mathbf{B}^-_{\gamma\,\text{rw}}}$. According to Lemma 5.1 (c), L_{mix} and $L_{B^+_{\gamma\,\text{sym}}} + D_{B^+_\gamma}^{-\frac{1}{2}} Q_{B^-_\gamma} D_{B^+_\gamma}^{-\frac{1}{2}}$ have the same eigenvalues (see also Remark 5.2) and if v is an eigenvector of the latter matrix, then $D_{B^+_\gamma}^{-\frac{1}{2}} v$ is an eigenvector of L_{mix}.

  We use (51) to compute approximations $\bar{B}^+_\gamma$ and $\bar{B}^-_\gamma$ of B^+_γ and B^-_γ, respectively. Then Remark 6.2 allows us to find approximations $\bar{D}_{B^+_\gamma}$ and $\bar{D}_{B^-_\gamma}$ of the degree matrices $D_{B^+_\gamma}$ and $D_{B^-_\gamma}$, respectively. Hence,

$$\begin{aligned} L_{B^+_{\gamma\,\text{sym}}} + D_{B^+_\gamma}^{-\frac{1}{2}} Q_{B^-_\gamma} D_{B^+_\gamma}^{-\frac{1}{2}} &= I - D_{B^+_\gamma}^{-\frac{1}{2}} B^+_\gamma D_{B^+_\gamma}^{-\frac{1}{2}} + D_{B^+_\gamma}^{-\frac{1}{2}} Q_{B^-_\gamma} D_{B^+_\gamma}^{-\frac{1}{2}} \\ &= I - D_{B^+_\gamma}^{-\frac{1}{2}} \left(B^+_\gamma - Q_{B^-_\gamma}\right) D_{B^+_\gamma}^{-\frac{1}{2}} \\ &= I - D_{B^+_\gamma}^{-\frac{1}{2}} \left(B^+_\gamma - B^-_\gamma - D_{B^-_\gamma} + D_{B^+_\gamma} - D_{B^+_\gamma}\right) D_{B^+_\gamma}^{-\frac{1}{2}} \\ &= 2I - D_{B^+_\gamma}^{-\frac{1}{2}} \left(B^+_\gamma - B^-_\gamma - D_{B^-_\gamma} + D_{B^+_\gamma}\right) D_{B^+_\gamma}^{-\frac{1}{2}} \\ &\approx 2I - (\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}} \left(\bar{B}^+_\gamma - \bar{B}^-_\gamma - \bar{D}_{B^-_\gamma} + \bar{D}_{B^-_\gamma}\right)(\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}} \\ &= 2I - \tilde{Q}\Upsilon\Sigma(\tilde{Q}\Upsilon)^T = \tilde{Q}\Upsilon\big[2I - \Sigma\big](\tilde{Q}\Upsilon)^T, \end{aligned}$$

  where we used (52) on $(\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}} \left(\bar{B}^+_\gamma - \bar{B}^-_\gamma - \bar{D}_{B^-_\gamma} + \bar{D}_{B^-_\gamma}\right)(\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}}$ to obtain the decomposition.

  Thus, the approximate eigenvalues of L_{mix} are obtained from $2I - \Sigma$, while the columns of $(\bar{D}^\dagger_{B^+_\gamma})^{\frac{1}{2}} \tilde{Q}\Upsilon$ give the approximate eigenvectors.

In each case, by choosing $k \geq m$ we can use the m leading approximate eigenvalues computed by the Nyström method (and their corresponding approximate eigenvectors) as approximations for the m leading eigenvalues of L_{mix} (and corresponding eigenvectors).

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

7. Numerical studies

This section presents the results of numerical studies for a variety of examples. All algorithms are implemented in Python 3.8. For each example, we selected one value of the parameter γ and then we tested our MMBO algorithms for the six cases of L_{mix} in (38). Because we observed that $L_{\text{mix}} = L_W + \gamma Q_P$ and $L_{\text{mix}} = L_{B_\gamma^+} + Q_{B_\gamma^-}$ always gave the worst modularity scores, the results for these two cases are not presented for the sake of brevity. To save space, in our tables we indicate the remaining four choices of L_{mix} by the main matrices on which they depend: '$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$', '$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$', '$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$' and '$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$'.

Our MMBO schemes are compared to the modularity MBO algorithm from Hu *et al.* [33], Boyd *et al.*'s pseudospectral balanced TV method [7], the CNM approach as given in Clauset *et al.* [19], the Louvain method from Blondel *et al.* [6], the Leiden algorithm introduced by Traag *et al.* [71] and spectral clustering as in Shi and Malik [37].[37] The functions for the CNM method, the Louvain method, the Leiden algorithm and spectral clustering can be called directly in the Python libraries `NetworkX`[38] [68], `leidenalg` [70] and `scikit-learn` [59]. We assess the results not only based on their modularity scores and computing times, but also according to the other metrics that we present in Section 7.1.3. Our simulations were performed on a MacBook Air (13-inch, 2017), with a 1.8 GHz Dual-Core Intel Core i5 processor and 8 GB 1600MHz DDR3 memory.

7.1. Related algorithms, null model and additional evaluation metrics

7.1.1. Related algorithms

The method from Hu *et al.* [33] is based on the observation that maximising modularity $\mathcal{Q}_\gamma(\mathcal{A}; W, P^{\text{NG}})$[39] over all partitions $\mathcal{A}$ with K (possibly empty) parts is equivalent to minimising

$$\mathcal{Q}_\gamma^{\text{Hu}}(U) := \mathcal{TV}_W(U) - \gamma \sum_{k=1}^{K} \sum_{i \in V} (d_W)_i \left(U_{ik} - \overline{U}_{ik}\right)^2 = \mathcal{TV}_W(U) - \gamma \langle U - \overline{U}, U - \overline{U} \rangle_W \tag{53}$$

over all $U \in Pt_0(K)$, where, in analogy to (23), we define

$$Pt_0(K) := \left\{ U \in \mathbb{R}^{|V| \times K} : \forall i \in V \ \ \forall l \in \{1, \dots, K\} \ \ U_{il} \in \{0, 1\} \text{ and } \sum_{k=1}^{K} U_{ik} = 1 \right\}.$$

In addition, we define for all $U \in \mathbb{R}^{|V| \times K}$ the mean $\overline{U} \in \mathbb{R}^{|V| \times K}$ by

$$\overline{U}_{ik} := \frac{1}{\text{vol}_W(V)} \sum_{j \in V} (d_W)_j U_{jk},$$

and we recall the inner product $\langle \cdot, \cdot \rangle_W$ from (35). As per Section 6.4, $\mathbf{1}_{|V|}$ is the column vector in $\mathbb{R}^{|V|}$ which has each entry equal to 1. To unclutter the notation, in this subsection, we simplify this notation to $\mathbf{1}$. Then $d_W = D_W \mathbf{1}$ is the column vector with entries $(d_W)_i$ and

$$\overline{U} = \frac{1}{\text{vol}_W(V)} \mathbf{1} d_W^T U.$$

Similarly to what we have presented in the current paper, this observation in [33] has led Hu *et al.* to an MBO-type algorithm for modularity optimisation. Since the functional $\mathcal{Q}_\gamma^{\text{Hu}}$ in (53) is non-convex

[37] We note that we do not compare with the recent proximal gradient method from Sun and Chang [67], primarily because that method focuses on partitioning into two communities only.

[38] In an earlier draft, we used `python-louvain` [3] to implement Louvain's method (see Section 7.1.1), but we found it to be slower than `NetworkX`, despite yielding similar results in terms of modularity score and other evaluation metrics (see Section 7.1.3). Therefore, we include only results obtained by using `NetworkX` in our tables and figures. We thank one of the anonymous reviewers for suggesting that we seek a faster implementation.

[39] Thus, with the NG null model, as in (16).

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

as argued in [7, Footnote 2],[40] a convex splitting approach is needed to solve the linear-dynamics step of the MBO-type scheme. The operator used in this step (in the place where our algorithm uses L_{mix}) is given by[41]

$$L_{\text{Hu}}U := L_W U - 2\gamma D_W(U - \overline{U}) \quad \text{so that} \quad L_{\text{Hu}} = L_W + \frac{2\gamma}{\text{vol}_W(V)} d_W d_W^T - 2\gamma D_W,$$

where the second expression follows since $D_W\overline{U} = \frac{1}{\text{vol}_W(V)} D_W \mathbf{1} d_W^T U = \frac{1}{\text{vol}_W(V)} d_W d_W^T U$.

For the actual computation of the linear-dynamics step, a projection onto the m leading eigenvectors of the graph Laplacian is used.

As can be seen from L_{Hu}, in [33] the unnormalised graph Laplacian L_W is used in the algorithm. This form of L_{Hu} is obtained by using the inner product from (34) in the derivation of the gradient flow of (53) (see footnote 41). If the inner product from (35) is used instead, we obtain the operator

$$L_{\text{Hu,rw}}U := L_{W,\text{rw}}U - 2\gamma(U - \overline{U}) \quad \text{so that} \quad L_{\text{Hu,rw}} = L_{W,\text{rw}} + \frac{2\gamma}{\text{vol}_W(V)} \mathbf{1} d_W^T - 2\gamma I.$$

For the symmetrically normalised variant, we use[42]

$$L_{\text{Hu,sym}} := L_{W,\text{sym}} + \frac{2\gamma}{\text{vol}_W(V)} d_W^{\frac{1}{2}} (d_W^{\frac{1}{2}})^T - 2\gamma I,$$

where $d_W^{\frac{1}{2}} = D_W^{\frac{1}{2}} \mathbf{1}$ denotes the column vector with entries $(d_W)_i^{\frac{1}{2}}$.

We also present tests for Hu *et al.*'s method with these normalised operators, because, as in the MMBO algorithms above, we obtained better modularity scores (and better scores on most if not all other measures that we introduce in Section 7.1.3) using those operators.

In [33], an Euler finite-difference scheme is used for Hu *et al.*'s method which requires a choice of time step along the same lines as our $\delta t := N_t^{-1}\tau$ in Sections 5.2.2 and 5.4.2. We do not select this time step by using the method from Section 6.1 to select a value of τ, since the operators L_{Hu}, $L_{\text{Hu,sym}}$ and $L_{\text{Hu,rw}}$ are not positive definite which poses problems for τ_{upp} in part (d) of Lemma 6.1.

Instead, we tested three different values for δt[43] Our first choice is $\delta t = 1$, following the choice in [33]. While this works for the unnormalised operator L_{Hu}, it is unsuitable for the normalised operators $L_{\text{Hu,sym}}$ and $L_{\text{Hu,rw}}$, as we empirically observe that they can have eigenvalues close to -1 in some cases (see Figures 1a, 5a and 5b). This causes $I + \delta t\Lambda$ in (43) and (46) to be (close to) singular, where Λ is the diagonal matrix containing eigenvalues of L_{Hu}, $L_{\text{Hu, sym}}$ or $L_{\text{Hu,rw}}$.

Our second choice is $\delta t = 1/N_t$, which looked like a good choice in preliminary simulations; at least it gave better results than $\delta t = 1$.

Our third choice is $\delta t = \chi_{\text{Hu}}/\lambda_{\text{Hu}}$, where λ_{Hu} is the greatest eigenvalue for the unnormalised operator L_{Hu} (i.e., the smallest in absolute value if the eigenvalues are negative) and χ_{Hu} is the greatest eigenvalue for either L_{Hu}, $L_{\text{Hu,rw}}$ or $L_{\text{Hu,sym}}$. In the case of the unnormalised operator L_{Hu}, $\lambda_{\text{Hu}} = \chi_{\text{Hu}}$ and we are back to the choice from [33]: $\delta t = 1$. For the two normalised operators, in this choice the time step is scaled

[40] More accurately, the functional is still non-convex, even if the non-convex domain $Pt_0(K)$ is relaxed to be the convex domain $\mathbb{R}^{|V|\times K}$. Depending on the value of γ, this non-convexity can still persist if $\mathcal{TV}_W(U)$ is replaced by the graph Dirichlet energy. Since this replacement gives the functional that generates, via arguments analogous to those in Sections 4 and 5, to the linear dynamics step of the corresponding MBO scheme, non-convexity of this functional suggests the use of a convex splitting scheme to solve the linearly dynamics step.

[41] The derivation is similar to the way that we, for example, derived a gradient flow in (29) and then a corresponding MBO scheme in Section 5.1. The identities $\overline{\overline{U}} = \overline{U}$ and $\langle U, \overline{V}\rangle_W = \langle \overline{U}, V\rangle_W$, for all matrices $U, V \in \mathbb{R}^{|V|\times K}$, and their corollary $\langle U - \overline{U}, \overline{V}\rangle_W = 0$, are useful in this and following computations.

[42] As per footnote 32, a symmetrically normalised variant $L_{\text{Hu,sym}}$ cannot be obtained from (53) in the same way that we derived L_{Hu} and $L_{\text{Hu,rw}}$. Instead, inspired by footnote 22, we replace each instance of U in (53) by $D_W^{-\frac{1}{2}}U$ and then follow the usual recipe with the inner product from (34). We emphasise that $\overline{D_W^{-\frac{1}{2}}U}$ should be used in the second term of (53), not $D_W^{-\frac{1}{2}}\overline{U}$. To understand the second term in $L_{\text{Hu,sym}}$, we observe that $D_W^{\frac{1}{2}}\overline{D_W^{-\frac{1}{2}}U} = \frac{1}{\text{vol}_W(V)} D_W^{\frac{1}{2}} \mathbf{1}\mathbf{1}^T D_W D_W^{-\frac{1}{2}} U = \frac{1}{\text{vol}_W(V)} d_W^{\frac{1}{2}} (d_W^{\frac{1}{2}})^T U$.

[43] Or τ_n or dt in the notation of [33].

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

according to the ratio of eigenvalues to avoid the singular operator problem we noted earlier. Based on our numerical experiments, this third choice for δt consistently outperforms the other two in terms of modularity score, computation time and other evaluation metrics (see Section 7.1.3). Therefore, we present only the results obtained via this third choice in all the tables and figures below that involve any of the variants of Hu *et al.*'s method.

The method also requires the maximal[44] number of non-empty clusters K as input. In [33], the algorithm is run for multiple values of K and the optimal output is selected, whereas we will run the algorithm at specifically selected values of K so that all algorithms we test can be compared at the same value(s) of K.

As shorthand, we may call this method Hu's method.

Similar to the method from [33], also the method by Boyd *et al.* [7] is based on finding a functional whose minimisation over $Pt_0(K)$ is equivalent to the maximisation of $\mathcal{Q}_\gamma(\mathcal{A}; W, P^{\mathrm{NG}})$ over all partitions $\mathcal{A}$ with K (possibly empty) parts:

$$\mathcal{Q}_\gamma^{\mathrm{Boyd}}(U) := \mathcal{TV}_W(U) + \frac{\gamma}{\mathrm{vol}_W(V)} \sum_{k=1}^{K} \left(\sum_{j \in V} (d_W)_j U_{jk} \right)^2 = \mathcal{TV}_W(U) + \gamma \langle \overline{U}, \overline{U} \rangle_W. \tag{54}$$

Different from [33], this functional is convex.[45] Based on this equivalence, also in [7] an MBO-type algorithm is proposed, analogous to what we did in Sections 4 and 5. The matrix which is used in the linear-dynamics step (in the place where we use L_{mix}) is

$$L_{\mathrm{Boyd}} := L_W + \frac{2\gamma}{\mathrm{vol}_W(V)} d_W d_W^T.$$

This is obtained from $\mathcal{Q}_\gamma^{\mathrm{Boyd}}$ in (54) in the standard way, using the inner product in (34) for the gradient flow. Indeed, taking into account the corollary given at the end of footnote 41, we find that[46]

$$\mathcal{Q}_\gamma^{\mathrm{Boyd}}(U) - \mathcal{Q}_\gamma^{\mathrm{Hu}}(U) = \gamma \langle U, U \rangle_W = \gamma \langle D_W U, U \rangle \tag{55}$$

and thus $L_{\mathrm{Boyd}} = L_{\mathrm{Hu}} + 2\gamma D_W$.

Similarly, if we use the inner product from (35) to obtain $L_{\mathrm{Boyd,rw}}$, then

$$L_{\mathrm{Boyd,rw}} := L_{\mathrm{Hu,rw}} + 2\gamma I = L_{W,\mathrm{rw}} + \frac{2\gamma}{\mathrm{vol}_W(V)} \mathbf{1} d_W^T.$$

Following a recipe similar to that in footnote 42, we replace U in (55) by $D_W^{-\frac{1}{2}}$ and thus in particular $\langle U, U \rangle_W$ by

$$\langle D_W^{-\frac{1}{2}} U, D_W^{-\frac{1}{2}} U \rangle_W = \langle D_W^{\frac{1}{2}} U, D_W^{-\frac{1}{2}} U \rangle = \langle U, U \rangle.$$

Computing its gradient according to the inner product in (34), we find

$$L_{\mathrm{Boyd,sym}} := L_{\mathrm{Hu,sym}} + 2\gamma I = L_{W,\mathrm{sym}} + \frac{2\gamma}{\mathrm{vol}_W(V)} d_W^{\frac{1}{2}} (d_W^{\frac{1}{2}})^T.$$

We use the pseudospectral balanced TV MBO scheme from [7], which uses a similar (truncated) eigendecomposition of the operator in the computation of the linear-dynamics step as our Algorithm 1.

As we did for Hu's method, however, and for the same reasons, we present tests for Boyd *et al.*'s method using normalised variants of L_{Boyd}.

---

[44] Also in this method empty clusters are allowed in the output, as will also be the case in Boyd *et al.*'s method which we describe later in this section.

[45] Similar to footnote 40, a more accurate statement is that the functional becomes convex if the non-convex domain $Pt_0(K)$ is replaced by the convex domain $\mathbb{R}^{|V| \times K}$.

[46] If $U \in Pt_0(K)$ encodes the partition $\mathcal{A} = \{A_l\}_{l=1}^K$, that is, $U_{il} = 1$ if and only if $i \in A_l$ and $U_{il} = 0$ otherwise, then $\langle U, U \rangle_W = \sum_{k=1}^K \sum_{i \in V} (d_W)_i U_{ik}^2 = \sum_{k=1}^K \sum_{i \in A_k} (d_W)_i = \sum_{k=1}^K \mathrm{vol}_W(A_k) = \mathrm{vol}_W(V)$. Thus, for the purpose of maximisation over $Pt_0(K)$, the difference between $\mathcal{Q}_\gamma^{\mathrm{Hu}}$ and $\mathcal{Q}_\gamma^{\mathrm{Boyd}}$ is a constant, and therefore irrelevant, term. Considering these functionals on all of $\mathbb{R}^{|V| \times K}$, however, this term is no longer constant and leads to the observed convexity of $\mathcal{Q}_\gamma^{\mathrm{Boyd}}$ on $\mathbb{R}^{|V| \times K}$ (see footnote 45).

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

As discussed in Section 6.1, [7] proposes a selection method for τ. We use that method for determining τ in Boyd *et al.*'s method. Specifically, we use the value suggested by [7, Propositions 4.1 and 4.2], which are analogous to our Lemma 6.1. The method requires that the maximum number of non-empty clusters K is known in advance.

To shorten the name, we may call this method Boyd's method.

The CNM method from Clauset *et al.* [19], Louvain method from Blondel *et al.* [6] and Leiden algorithm from Traag *et al.* [71] are all greedy methods, which iteratively update communities in a way that maximises the immediate increase in modularity. All methods differ in the types of updates they perform. In all methods, initially each node in the network is considered as a separate community and then nodes may be moved to different communities to increase modularity in a local movement phase. The CNM method sequentially computes for each community (according to some ordering of the communities) and each of its neighbours the change in modularity if those two communities would merge. If any increase in modularity is possible, one of the mergers that produces the maximum increase is performed. This continues iteratively until no merger of neighbouring communities increases modularity. The Louvain method sequentially computes for each node (according to some ordering of the nodes) and for each of its neighbours the change in modularity if the node were to be removed from its current community and assigned to its neighbour's community. If any increase in modularity is possible, one of the assignments that produces the maximum increase is performed. This continues iteratively until no more increase in modularity is possible in this way. Then the first phase of the algorithm ends and, for the CNM and Louvain algorithms, the second phase, an aggregation phase starts, in which a new graph is built that contains a node for each community obtained in the first previous phase. The edge weight between two nodes in the new graph is obtained by adding together all weights of the edges that connect the corresponding communities in the original network. To this new graph, the procedure of the first phase is again applied. The two phases are iterated in this way until no more change in modularity is obtained.

The Louvain algorithm can lead to poorly connected communities [71]. The Leiden algorithm improves upon the Louvain algorithm through an additional refinement phase that takes place between the local movement and aggregation phases. It ensures that every community is internally connected. Each iteration of the Leiden algorithm includes three phases: local movement of nodes, refinement of communities to ensure internal connectivity and aggregation of the network. This results in more accurate and stable community detection.

The local movement phase of the Leiden algorithm is similar to that of the Louvain algorithm, yet more efficient as it only (re)visits those nodes whose neighbours' community assignments have changed as a result of an earlier local move, rather than all nodes. This contributes to the Leiden algorithm generally being faster, particularly for large networks. In the refinement phase communities from the unrefined partition that has been formed in the local movement phase can split further into subsets that are better connected internally. For each community from the unrefined partition, a refined partition is initialised by assigning each node to its own community. Then singleton communities are merged with other communities according to a connectivity-based probability distribution. Importantly, these mergers only happen within each community from the unrefined partition so that the communities that were found in the local movement phase can split, but not merge. In the aggregation phase of the Leiden algorithm, the aggregated graph is based on the refined partition, yet the partition of this new graph, with which the local movement phase of the next iteration is initiated, is based on the unrefined partition that resulted from the previous local movement phase, rather than the refined partition. This means that in the Leiden algorithm, contrary to the Louvain algorithm, not all communities of the initial partition at the start of each new local movement phase need to be singletons (except in the first iteration of the algorithm). For a detailed description of the Leiden algorithm, we refer to [71, Supplementary Information].

The stopping criterion for the CNM, Louvain and Leiden methods is based on the change in modularity score, or rather, the lack of increase in this score among all community update options that are available to the algorithm in question. This is different from the stopping condition (48), which depends

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

on the Euclidean change in subsequent partition matrices found by the algorithm, rather than their modularity scores. The nature of the CNM, Louvain and Leiden methods makes it difficult to introduce a stopping condition like (48). This is an important reason why we also consider the modularity-based stopping criterion in (49). Another motivation is the observation that often the modularity score improves only slowly after the first few iterations of the algorithms, as we will see in some more detail later. Because of this difference in stopping conditions, we present the results for the CNM, Louvain and Leiden methods in different tables than the results for the MMBO schemes, Hu's method and Boyd's method.

Furthermore, the CNM, Louvain and Leiden methods do not require the (maximum) number of clusters K as input, in contrast to Hu's, Boyd's and our methods; rather, K is one of the outputs of these methods, implicitly given by the final clustering when the stopping condition has been reached in the CNM, Louvain and Leiden methods. By construction, these methods do not output empty clusters.

Spectral clustering proposed by Shi and Malik [37] is a clustering approach based on spectral graph theory (see also von Luxburg [74]). It was not developed for optimising modularity specifically, but we use it in our tests to compare how our method compares to it in the various performance measures that we consider. The basic idea is to embed the data, represented as nodes in a graph, into Euclidean space based on the K leading eigenvectors resulting from the eigendecomposition of the random walk Laplacian L_{rw}. Clusters are then found by using a clustering algorithm that can be applied to points in Euclidean space. We use the commonly used K-means algorithm, as part of the `SpectralClustering` function in the Python `scikit-learn` library [59]. The maximum[47] number of non-empty clusters K needs to be specified in advance.

7.1.2. Null model for modularity optimisation

Hu's method, Boyd's method, the CNM method, the Louvain method and the Leiden algorithm all are based on modularity with the NG null model.[48]

Therefore, in our examples, we also employ the NG null model in our MMBO algorithms. We refer to Section 2.3 for more details. In particular, we wish to optimise $\mathcal{Q}(\mathcal{A}; W, P^{\mathrm{NG}})$ from (16). This is the quantity reported in the tables in this section as 'NG modularity'.

7.1.3. Additional evaluation metrics

The main goal of our numerical tests is to find out how well our method performs on the modularity maximisation task. As a secondary goal, we also want to compare the clusters (communities) that we obtain with ground truth communities in cases where such ground truth is available. For this comparison, we employ a number of different evaluation metrics:[49] *purity* [63], *inverse purity*, the *adjusted rand index* (ARI) [25] and *normalised mutual information* (NMI) [40, 42]. For clarity's sake, we will call the communities that are obtained by the algorithm *clusters* and the communities that are present in the ground truth *classes*.

Let $\mathcal{C}$ and $\mathcal{C}'$ be partitions of V, where $\mathcal{C} := \{C_1, \dots, C_K\}$ is the set of K clusters to be evaluated and $\mathcal{C}' := \{C'_1, \dots, C'_{K'}\}$ is the set of K' classes of the ground truth. We assume here that all clusters and all classes are non-empty.

Purity is an evaluation metric that quantifies the proportion of nodes in a cluster that are members of the same majority class and is defined as

$$\mathrm{Purity}(\mathcal{C}, \mathcal{C}') := \frac{1}{|V|} \sum_{k=1}^{K} \max_{1 \le l \le K'} \left| C_k \bigcap C'_l \right| .$$

[47] The K-means algorithm can return empty clusters.

[48] Since spectral clustering is not designed to be a modularity optimisation method, it does not rely on a choice of null model.

[49] Or evaluation measures, where we do not necessarily intend either 'metric' or 'measure' to be read in their strict mathematical meaning.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Purity achieves its maximum value of 1 if $\mathcal{C} = \mathcal{C}'$, but this maximiser is not unique: if $\mathcal{C}$ consists of $|V|$ singleton clusters, this maximum value is also obtained. A high purity can thus be achieved artificially by having many clusters, since this metric does not penalise cluster size.

We note that purity is not symmetric in $\mathcal{C}$ and $\mathcal{C}'$. By interchanging both arguments, we obtain *inverse purity:*

$$\text{InvPurity}(\mathcal{C}, \mathcal{C}') := \text{Purity}(\mathcal{C}', \mathcal{C}) = \frac{1}{|V|} \sum_{l=1}^{K'} \max_{1 \le k \le K} \left| C_k \bigcap C_{l'} \right|.$$

Inverse purity is biased in favour of large clusters. In particular, the maximum value 1 is not only obtained if $\mathcal{C} = \mathcal{C}'$, but also if all nodes are grouped together into a single cluster.

Purity and inverse purity both quantify the number of correctly clustered nodes, under some definition of 'correct'. The next metrics we list quantify numbers of correctly clustered *pairs* of nodes.

We classify a pair of distinct nodes $i, j \in V$ as *true positive* if they belong to the same cluster in $\mathcal{C}$ and the same class in $\mathcal{C}'$, that is, if there are k and k', such that $i, j \in C_k \cap C'_{k'}$. We denote the total number of true positives by *TP*. Similarly, a pair of distinct nodes i, j forms a *true negative*, if both nodes are in distinct clusters and distinct classes, that is, if there are distinct k and l and distinct k' and l' such that $i \in C_k \cap C'_{k'}$ and $j \in C_l \cap C'_{l'}$. The total number of true negatives is *TN*. The pair i, j forms a *false positive*, if there are k and distinct k' and l', such that $i \in C_k \cap C'_{k'}$ and $j \in C_k \cap C'_{l'}$, and it forms a *false negative* if there are k' and distinct k and l such that $i \in C_k \cap C'_{k'}$ and $j \in C_l \cap C'_{k'}$. The total numbers of false positives and false negatives are denoted by FP and FN, respectively. These quantities are computed as follows:

$$\text{TP} := \sum_{k=1}^{K} \sum_{l=1}^{K'} \binom{|C_k \bigcap C_{l'}|}{2}, \quad \text{FP} := \sum_{k=1}^{K} \binom{|C_k|}{2} - \text{TP},$$

$$\text{FN} := \sum_{l=1}^{K'} \binom{|C_{l'}|}{2} - \text{TP}, \quad \text{TN} := \binom{|V|}{2} - \text{TP} - \text{FP} - \text{FN}.$$

The *Rand index* (RI) [61] is the proportion of correctly clustered node pairs:

$$\text{RI} := \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} = \binom{|V|}{2}^{-1} (\text{TP} + \text{TN}).$$

One of the drawbacks of RI is that it it does not consider the possibility of a coincidental agreement between the two partitions. The number and sizes of the clusters in each partition, as well as the total number of nodes, impact on the number of agreements of two partitions that can be expected to occur by chance. To mitigate this problem, the ARI is proposed in Hubert and Arabie [35] to be

$$\text{ARI} := \frac{\text{RI} - \mathbb{E}(\text{RI})}{\max \text{RI} - \mathbb{E}(\text{RI})}, \tag{56}$$

where the expected Rand index $\mathbb{E}(\text{RI})$ is computed based on the assumption that the contingency table with entries $|C_k \cap C_{l'}|$ is drawn from a generalised hypergeometric distribution[50] and we know that $\max \text{RI} = 1$. Hence ARI is well defined, unless $\mathbb{E}(\text{RI}) = 1$.[51] Various equivalent expressions for the ARI

[50] This translates to $\mathcal{C}$ and $\mathcal{C}'$ being random partitions conditioned on K, K' and all cluster and class sizes $|C_k|$ and $|C_{l'}|$ being fixed at their actual observed values. Under this assumption, it can be computed that $\mathbb{E}(\text{RI}) = 1 + 2\binom{|V|}{2}^{-2}(\text{TP} + \text{FN})(\text{TP} + \text{FP}) - \binom{|V|}{2}^{-1}(2\text{TP} + \text{FP} + \text{FN})$. See [35].

[51] From the third expression in (57) below, it can be seen that the denominator equals zero if and only if $\text{TP} = \text{FN} = \text{FP} = 0$ or $\text{FN} = \text{TN} = \text{FP} = 0$. The former case occurs if and only if $\text{TN} = \binom{|V|}{2}$, thus if and only if $K = K' = 1$, whereas the latter case occurs if and only if $\text{TP} = \binom{|V|}{2}$, thus if and only if $K = K' = |V|$. See also [15].

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

can be calculated:[52]

$$\begin{aligned}\mathrm{ARI} &= \frac{\binom{|V|}{2}\mathrm{TP} - (\mathrm{TP}+\mathrm{FN})(\mathrm{TP}+\mathrm{FP})}{\frac{1}{2}\binom{|V|}{2}(2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}) - (\mathrm{TP}+\mathrm{FN})(\mathrm{TP}+\mathrm{FP})} \\ &= \frac{\binom{|V|}{2}(\mathrm{TP}+\mathrm{TN}) - [(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN}) + (\mathrm{FN}+\mathrm{TN})(\mathrm{FP}+\mathrm{TN})]}{\binom{|V|}{2}^2 - [(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN}) + (\mathrm{FN}+\mathrm{TN})(\mathrm{FP}+\mathrm{TN})]} \\ &= 2\frac{\mathrm{TP}\cdot\mathrm{TN} - \mathrm{FN}\cdot\mathrm{FP}}{(\mathrm{TP}+\mathrm{FN})(\mathrm{FN}+\mathrm{TN}) + (\mathrm{TP}+\mathrm{FP})(\mathrm{FP}+\mathrm{TN})}. \end{aligned} \tag{57}$$

From (56), we see that the maximum value of the ARI is 1, which is obtained if $\mathrm{RI} = \max \mathrm{RI} = 1$[53] In Chacón and Rastrojo [15], the minimum possible value of the ARI is proven to be $-\frac{1}{2}$.[54]

The *entropy* of the clustering $\mathcal{C}$ is defined[55] to be[56]

$$H(\mathcal{C}) := -\sum_{k=1}^{K} \frac{|C_k|}{|V|} \log_2\left(\frac{|C_k|}{|V|}\right).$$

If we view the cluster assignment in $\mathcal{C}$ of a given node as a random variable with possible outcomes $k \in \{1, \ldots, K\}$ with uniform probability $\frac{|C_k|}{|V|}$, then $H(\mathcal{C})$ is the entropy associated with this random variable. The *joint entropy* of $\mathcal{C}$ and $\mathcal{C}'$ is

$$\text{joint } H(\mathcal{C}, \mathcal{C}') := -\sum_{k=1}^{K}\sum_{l=1}^{K'} \frac{|C_k \bigcap C'_l|}{|V|} \log_2\left(\frac{|C_k \bigcap C'_l|}{|V|}\right).$$

If we let our intuition be that (joint) entropy is a measure for the uncertainty associated with the cluster assignment of a node, then *mutual information* (MI) evaluates the reduction in this uncertainty that we obtain by considering $\mathcal{C}$ and $\mathcal{C}'$ jointly, rather than separately, that is,

$$\mathrm{MI}(\mathcal{C}, \mathcal{C}') := H(\mathcal{C}) + H(\mathcal{C}') - \text{joint } H(\mathcal{C}, \mathcal{C}') = \sum_{k=1}^{K}\sum_{l=1}^{K'} \frac{|C_k \bigcap C'_l|}{|V|} \log_2\left(\frac{|V||C_k \bigcap C'_l|}{|C_k||C'_l|}\right).$$

[52]The first expression below follows directly from [35, Formula (5)], the second one appears in Steinley [66, Formula (9)] and Chacón and Rastrojo [15], and the third expression appears in the the ARI function of the Python `scikit-learn` library [59] that we have used.

[53]If not simultaneously $\mathbb{E}(\mathrm{RI}) = 1$.

[54]In some sources in the literature, such as [53] one finds the (correct) claim that ARI has values in $[-1, 1]$. Indeed, it is not difficult to show that $\mathrm{ARI} < -1$ leads to the contradiction $(\mathrm{TP}+\mathrm{FP}+\mathrm{FN})\mathrm{TN} + \mathrm{TP}(\mathrm{FP}+\mathrm{FN}+\mathrm{TN}) + (\mathrm{FP}-\mathrm{FN})^2 + 2\mathrm{TP}\cdot\mathrm{TN} < 0$. Some sources, such as [65] even claim that the value -1 can be achieved, which contradicts the result in [15]. In an attempt to corroborate that the value -1 is in fact achievable, one could be tempted to choose $\mathrm{TP} = \mathrm{TN} = 0$ and maximise the resulting expression, which leads to $\mathrm{FP} = \mathrm{FN}$. However, we will show now that is not possible to construct $\mathcal{C}$ and $\mathcal{C}'$ that satisfy both $\mathrm{TP} = \mathrm{TN} = 0$ and $\mathrm{FP} = \mathrm{FN}$.

It follows from footnote 51, that ARI is undefined if $|V| = 1$. Assume $|V| \geq 2$. If the ground truth has $|V|$ classes, then $\mathrm{TN} = 0$ forces $\mathcal{C}$ not to contain any singletons. Thus, there is a cluster in $\mathcal{C}$ containing at least two elements, hence $\mathrm{FP} \geq 1$. However, since $|\mathcal{C}'| = |V|$, $\mathrm{FN} = 0 < \mathrm{FP}$. If the ground truth has only one class, then $\mathrm{TP} = 0$ forces $\mathcal{C}$ to have $|V|$ singleton clusters, in which case $\mathrm{FN} = \binom{|V|}{2} > 0 = \mathrm{FP}$. Thus, the ground truth must have at least two and at most $|V| - 1$ classes, which rules out $|V| = 2$. Hence, assume $|V| \geq 3$. By the pigeonhole principle, the ground truth must contain a class containing at least two elements, say $C_{ab}{}' = \{a, b\}$. Since the ground truth has at least two classes, there is another class $C_c{}'$ containing at least one element $c \in C_c{}'$ distinct from a and b. Since $\mathrm{TP} = 0$, $\mathcal{C}$ has to contain two disjoint clusters C_a, C_b with $a \in C_a$ and $b \in C_b$. Because $\mathrm{TN} = 0$, $c \in C_a$ and $c \in C_b$. This is a contradiction.

[55]The convention is that the kth term is zero if $|C_k| = 0$.

[56]For definiteness, and to honour its origins in information theory in Shannon [64], we have chosen base 2 for the logarithm, but in the normalised mutual information of (58), which is the quantity we are ultimately interested in, any overall constant factors that would appear in H and MI under a different choice of base would cancel out.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 2. *MNIST: parameter settings for the Nyström extension and edge weights in (59) (left) and parameter setting of the MMBO scheme (right)*

Parameter	Value	Parameter	Value
k	500	N_t	5
σ	100	η	10^{-5}
		γ	1

By subadditivity of the joint entropy, MI is always non-negative. Moreover, it is zero if and only if the cluster assignments associated with $\mathcal{C}$ and $\mathcal{C}'$ are independent random variables. Out of the many variants and generalisations of mutual information, we choose to use the following *normalised mutual information*:

$$\mathrm{NMI}(\mathcal{C}, \mathcal{C}') := \frac{2\mathrm{MI}(\mathcal{C}, \mathcal{C}')}{H(\mathcal{C}) + H(\mathcal{C}')}. \tag{58}$$

7.2. MNIST

The MNIST database is a widely used data set in computer vision and machine learning [43]. It comprises 70,000 black-and-white images of handwritten digits ranging from 0 to 9, each image consisting of 28×28 pixels. The data set consists of pairs of handwritten digit images together with ground truth digit assignments. We aim to group images of different digits into distinct communities. We construct a graph in which each node represents an image, thus $|V| = 70,000$, and the weighted edges are based on feature vector similarity. To create these feature vectors associated with each image, we project the images (which are associated with vectors in $\mathbb{R}^{28\times 28}$ containing their pixel's greyscale values) onto 50 principal components as determined by a principal component analysis (PCA). For each node i (corresponding to an image), we thus obtain a feature vector $x_i \in \mathbb{R}^{50}$ containing the coordinates with respect to the first 50 principal components. We define the weight between distinct nodes i and j as

$$\omega_{ij} := \exp\left(-\frac{||x_i - x_j||_2^2}{\sigma}\right), \tag{59}$$

where σ by the user. In this example, we choose $\sigma = 100$. The choice of σ impacts the number of clusters K found by the Louvain method. Since we use that value to set the maximal number of clusters in the MMBO methods as well as in the methods from Hu *et al.* and Boyd *et al.,* this in turn affects the size of the matrices in those methods, such as U^0 in Algorithms 1 and 2, which itself impacts the run times. Tests with $\sigma = 50$, which are not shown in the tables and figures in this paper, did show fewer clusters and lower (absolute) run times for all methods. If we compare the results in Tables 3 and 4, with the results in Table 5, the difference in run times between the MMBO methods and the Louvain method is substantial. In our tests with $\sigma = 50$, this difference was much less pronounced, in some cases even absent. This suggests that the Louvain method is more sensitive to changes in σ than the MMBO methods.

Defining the weights by (59) implies $\omega_{ij} = 1$ if and only if nodes i and j are identical. Additionally, we choose the same N_t as Hu et al. in [33] to compare the results of MMBO with those from [33], that is, $N_t = 5$.[57] See Table 2 for the parameters used in the methods.

[57] In the notation of [33]: $\eta = 5$.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 3. *MNIST: average time per run for computing eigenvalues and eigenvectors, the average time per run for all MBO iterations, and the average number of MBO iterations per run for the MMBO schemes, Hu et al.'s method, and Boyd et al.'s method when using $m = 130$ and $K = 764$ and the partition-based stopping criterion from (48). The number of iterations is rounded to the nearest integer. The best average result in each column is shown in boldface*

Method		Avg. time for eigenvalues and eigenvectors (s)	Avg. time for MBO iterations (s)	Avg. # of iterations
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	14.4 ($\pm$ 2.9)	64.3 ($\pm$ 11.8)	134 ($\pm$ 36)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	13.7 ($\pm$ 2.0)	63.2 ($\pm$ 10.3)	127 ($\pm$ 38)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	15.1 ($\pm$ 4.4)	**19.4** ($\pm$ 3.8)	**55** ($\pm$ 21)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	15.6 ($\pm$ 4.8)	67.2 ($\pm$ 7.0)	91 ($\pm$ 25)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	14.4 ($\pm$ 2.9)	178.2 ($\pm$ 18.5)	152 ($\pm$ 36)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	13.7 ($\pm$ 2.0)	153.6 ($\pm$ 20.2)	127 ($\pm$ 30)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	15.1 ($\pm$ 4.4)	144.2 ($\pm$ 14.2)	119 ($\pm$ 18)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	15.6 ($\pm$ 4.8)	164.7 ($\pm$ 15.4)	140 ($\pm$ 23)
Hu *et al.*	$L_{W_{\text{sym}}}$	12.7 ($\pm$ 1.6)	186.2 ($\pm$ 26.4)	138 ($\pm$ 27)
	$L_{W_{\text{rw}}}$	13.5 ($\pm$ 1.2)	179.6 ($\pm$ 20.3)	130 ($\pm$ 16)
Boyd *et al.*	$L_{W_{\text{sym}}}$	**12.6** ($\pm$ 1.6)	199.4 ($\pm$ 20.5)	206 ($\pm$ 25)
	$L_{W_{\text{rw}}}$	13.5 ($\pm$ 1.2)	197.7 ($\pm$ 21.7)	187 ($\pm$ 34)

Table 4. *MNIST: average time per run for computing eigenvalues and eigenvectors, the average time per run for all MBO iterations, and the average number of MBO iterations per run for the MMBO scheme, Hu et al.'s method, and Boyd et al.'s method when using $m = 130$ and $K = 764$ and the modularity-based stopping condition from (49). The number of iterations is rounded to the nearest integer. The best average result in each column is shown in boldface*

Method		Avg. time for eigenvalues and eigenvectors (s)	Avg. time for MBO iterations (s)	Avg. # of iterations
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	14.4 ($\pm$ 2.9)	11.2 ($\pm$ 2.9)	32 ($\pm$ 4)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	13.7 ($\pm$ 2.0)	12.6 ($\pm$ 3.2)	36 ($\pm$ 6)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	15.1 ($\pm$ 4.4)	**9.3** ($\pm$ 2.5)	**17** ($\pm$ 4)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	15.6 ($\pm$ 4.8)	10.2 ($\pm$ 2.0)	28 ($\pm$ 4)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	14.4 ($\pm$ 2.9)	24.8 ($\pm$ 4.1)	36 ($\pm$ 5)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	13.7 ($\pm$ 2.0)	28.9 ($\pm$ 5.4)	34 ($\pm$ 7)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	15.1 ($\pm$ 4.4)	18.6 ($\pm$ 4.7)	25 ($\pm$ 7)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	15.6 ($\pm$ 4.8)	17.4 ($\pm$ 5.1)	32 ($\pm$ 6)
Hu *et al.*	$L_{W_{\text{sym}}}$	**8.1** ($\pm$ 1.2)	30.4 ($\pm$ 12.8)	27 ($\pm$ 10)
	$L_{W_{\text{rw}}}$	8.5 ($\pm$ 1.3)	71.9 ($\pm$ 13.5)	29 ($\pm$ 8)
Boyd *et al.*	$L_{W_{\text{sym}}}$	9.7 ($\pm$ 1.2)	34.9 ($\pm$ 10.2)	43 ($\pm$ 19)
	$L_{W_{\text{rw}}}$	9.8 ($\pm$ 1.3)	60.7 ($\pm$ 20.8)	39 ($\pm$ 21)

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 5. *MNIST: average performance of algorithms regarding modularity scores, various classification metrics, and average computation time per run under NG null model. The best average results in each column are shown in boldface (we exclude the ground truth numbers). For the number of non-empty clusters we consider the one closest to the ground truth to be 'best' in this context*

Method	NG modularity	ARI	Purity	Inv. purity	NMI	Time Time (s)	Non-empty clusters
Louvain	**0.58** ($\pm$ 0.01)	**0.69** ($\pm$ 0.01)	**0.78** ($\pm$ 0.01)	0.88 ($\pm$ 0.0)	**0.79** ($\pm$ 0.01)	1146.3 ($\pm$ 131.9)	764 ($\pm$ 23)
CNM	0.30 ($\pm$ 0.0)	0.19 ($\pm$ 0.0)	0.33 ($\pm$ 0.0)	**0.93** ($\pm$ 0.0)	0.45 ($\pm$ 0.0)	8196.4 ($\pm$ 13.3)	808 ($\pm$ 0)
Leiden	0.55 ($\pm$ 0.01)	0.67 ($\pm$ 0.07)	0.75 ($\pm$ 0.06)	0.88 ($\pm$ 0.06)	**0.79** ($\pm$ 0.03)	**11.2** ($\pm$ 0.6)	**659** ($\pm$ 2)
Ground truth	0.31	1.0	1.0	1.0	1.0	–	10

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 6. *MNIST: average performance of different algorithms regarding modularity scores, various classification metrics, and total computation time under NG null model when using* $m = 130$ *and* $K = 764$ *and the partition-based stopping criterion (48). The best average result in each column is shown in boldface. For the number of non-empty clusters we consider the one closest to the ground truth number* 10 *to be 'best' in this context*

Method	Method	NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.55 ($\pm$0.02)	0.44 ($\pm$0.07)	0.77 ($\pm$0.04)	0.52 ($\pm$0.07)	0.63 ($\pm$0.04)	80.3 ($\pm$14.8)	20 ($\pm$2)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.55 ($\pm$0.02)	0.45 ($\pm$0.09)	0.77 ($\pm$0.05)	0.53 ($\pm$0.08)	0.63 ($\pm$0.04)	76.3 ($\pm$13.5)	20 ($\pm$3)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.28 ($\pm$0.06)	0.29 ($\pm$0.02)	0.40 ($\pm$0.03)	0.78 ($\pm$0.02)	0.44 ($\pm$0.02)	**34.7** ($\pm$8.0)	**5** ($\pm$1)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.53 ($\pm$0.06)	0.45 ($\pm$0.10)	0.74 ($\pm$0.07)	0.56 ($\pm$0.06)	0.61 ($\pm$0.04)	84.3 ($\pm$10.9)	26 ($\pm$5)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	**0.56** ($\pm$0.02)	0.42 ($\pm$0.04)	0.77 ($\pm$0.04)	0.49 ($\pm$0.05)	0.63 ($\pm$0.02)	191.7 ($\pm$21.9)	22 ($\pm$2)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.55 ($\pm$0.02)	0.44 ($\pm$0.07)	0.78 ($\pm$0.03)	0.51 ($\pm$0.07)	0.63 ($\pm$0.03)	167.2 ($\pm$23.0)	21 ($\pm$2)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.55 ($\pm$0.03)	0.45 ($\pm$0.06)	0.75 ($\pm$0.05)	0.53 ($\pm$0.05)	0.63 ($\pm$0.03)	157.9 ($\pm$18.8)	18 ($\pm$2)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.54 ($\pm$0.02)	0.47 ($\pm$0.09)	0.78 ($\pm$0.04)	0.54 ($\pm$0.10)	0.64 ($\pm$0.04)	177.5 ($\pm$20.3)	19 ($\pm$3)
Hu *et al.*	$L_{W_{\text{sym}}}$	0.54 ($\pm$0.03)	0.43 ($\pm$0.05)	0.79 ($\pm$0.06)	0.49 ($\pm$0.04)	0.63 ($\pm$0.03)	198.4 ($\pm$26.8)	22 ($\pm$2)
	$L_{W_{\text{rw}}}$	0.54 ($\pm$0.02)	0.42 ($\pm$0.06)	0.79 ($\pm$0.08)	0.48 ($\pm$0.03)	0.63 ($\pm$0.03)	190.8 ($\pm$23.8)	22 ($\pm$2)
Boyd *et al.*	$L_{W_{\text{sym}}}$	0.51 ($\pm$0.05)	**0.60** ($\pm$0.14)	**0.80** ($\pm$0.15)	0.76 ($\pm$0.12)	**0.75** ($\pm$0.06)	201.4 ($\pm$22.5)	17 ($\pm$8)
	$L_{W_{\text{rw}}}$	0.52 ($\pm$0.03)	0.52 ($\pm$0.14)	0.72 ($\pm$0.11)	0.75 ($\pm$0.10)	0.71 ($\pm$0.04)	200.5 ($\pm$21.2)	14 ($\pm$5)
Spectral clustering		0.51 ($\pm$0.01)	0.43 ($\pm$0.05)	0.75 ($\pm$0.03)	**0.81** ($\pm$0.04)	0.69 ($\pm$0.06)	186.4 ($\pm$16.9)	764 ($\pm$0)

The values that we present in Tables 3–7 are averages computed over twenty runs for each algorithm, followed by the maximum deviation (in absolute value[58]) from each average in parentheses. It is worth noting that the initial condition U^0, which involves randomly assigning each node to a community (with each community containing at least one node) as outlined in Algorithms 1 and 2, is likely to differ with each run. However, this variation in initial conditions does not impact greatly the modularity score achieved by the algorithms.

Both the Louvain method and the Leiden algorithm achieve similar modularity; see Table 5. Although the running time of the Leiden algorithm is much faster than that of the Louvain method, the primary goal of this paper is to maximise modularity. Since the Louvain method achieves the highest modularity, we select the optimal K found by the Louvain method. The highest modularity obtained by the Louvain method is achieved around[59] $K = 764$. We also tested the MBO-based methods with $K = 659$, inspired by the results of the Leiden algorithm in Table 5. We obtained a lower modularity than for $K = 764$ in this way and also the values of the other classification metrics were lower. Therefore, we have not included those results in this paper, although the running time was also reduced.

For a fair comparison of the different methods' resulting modularity values, we wish to use the same upper bound K on the number of clusters for each method. We recall that for the Louvain method K is an output of the algorithm (with K being the exact number of non-empty clusters, not just an upper bound), whereas for the methods of Hu *et al.* and Boyd *et al.,* as well as for our MMBO method, K has to be given as input (and is only an upper bound on the number of non-empty clusters, since the methods may output empty clusters). Thus, we first use the Louvain approach to determine an appropriate value for K. Then we run the methods of Hu *et al.* and Boyd *et al.* and our MMBO method with the same value for K, namely $K = 764$.

Figure 1 displays the spectra of the first 180 eigenvalues for the different choices of L_{mix} under the NG null model as well as $L_{\text{Hu,sym}}$, $L_{\text{Hu,rw}}$, $L_{\text{Boyd,sym}}$ and $L_{\text{Boyd,rw}}$. In each of the panels in Figure 1, the two plotted curves overlap since the eigenvalues of the corresponding pairs of operators are identical, that is, the eigenvalues of $L_{\text{Hu,sym}}$ and $L_{\text{Hu,rw}}$ are identical, as are the eigenvalues of $L_{\text{Boyd,sym}}$ and $L_{\text{Boyd,rw}}$, the eigenvalues of $L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$ and $L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$, and also the eigenvalues of $L_{B^+_{\gamma,\text{sym}}} + Q_{B^-_{\gamma,\text{sym}}}$ and $L_{B^+_{\gamma,\text{rw}}} + Q_{B^-_{\gamma,\text{rw}}}$.

Using the MMBO scheme from Algorithm 1 as an illustration, Figure 2 demonstrates that the highest modularity is achieved if m, the number of leading eigenvalues that is used for the eigendecomposition of L_{mix}, is chosen at 130. Therefore, we pick $m = 130$.

First, we study the computing times of the two main parts of the algorithms. Then, we discuss the performance in terms of the modularity scores and other evaluation metrics, and its dependency on the choice of null model and the stopping criterion. Finally, the effect of using some *a priori* known node assignments will be explored.

There are two main parts to the execution time of the MMBO schemes, Hu *et al.*'s method and Boyd *et al.*'s method: the computation of the eigenvalues and eigenvectors of L_{mix} for MMBO or of the method-specific matrices for Hu and Boyd's methods, and the iteration of the linear-dynamics and thresholding steps. In order to speed up the computation of the eigenvalues and eigenvectors, we use the Nyström extension with QR decomposition as explained in Section 6.4. Table 3 presents the computing times if the partition-based stopping criterion from (48) is used. In Table 4, we show the computing times if the modularity-based stopping criterion from (49) is used instead. The use of the second stopping criterion is motivated by the observation that often the modularity score improves only slowly after the first few iterations. In Figure 3, for example, the modularity score changes little after the first 35 iterations, but the algorithm continues because we do not set any stopping criterion. This behaviour is similar to the

[58]To be explicit, the top-left entry 14.4($\pm$2.9) in Table 3, for example, indicates that the average time over 20 runs is 14.4 seconds and the time, or times, that deviate most from this average in all of the 20 runs are equal to $14.4 - 2.9$ seconds or to $14.4 + 2.9$ seconds.

[59]We implement the Louvain method using the `NetworkX` package for Python [68]. On the website [68], it is noted that the order in which nodes are considered can influence the final output, and in the algorithm, this order is determined by a random shuffle.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 7. *MNIST: average performance of algorithms regarding modularity scores, various classification metrics, and total computation time under NG null model with ('10%') and without ('no') 10% mild semi-supervision when using the modularity-based stopping condition (49). In both the unsupervised and mildly semi-supervised case, $m = 130$ and $K = 764$ are used. With mild semi-supervised clustering, $m = 130$ and $K = 764$ is used. The best average results with and without mild semi-supervision in each column are shown in boldface. For the number of non-empty clusters we consider the one closest to the ground truth number 10 to be 'best' in this context*

Method		GT	NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	no	0.55 (± 0.02)	0.43 (± 0.06)	0.77 (± 0.04)	0.47 (± 0.03)	0.62 (± 0.03)	23.0 (± 5.2)	21 (± 3)
		10%	0.31 (± 0.01)	0.81 (± 0.02)	0.90 (± 0.01)	0.90 (± 0.01)	0.79 (± 0.01)	**12.4** (± 4.1)	**10** (± 0)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	no	0.55 (± 0.02)	0.42 (± 0.04)	0.76 (± 0.04)	0.49 (± 0.04)	0.62 (± 0.01)	24.5 (± 8.2)	21 (± 3)
		10%	0.31 (± 0.01)	0.80 (± 0.02)	0.90 (± 0.01)	0.90 (± 0.01)	0.79 (± 0.01)	13.6 (± 3.5)	**10** (± 0)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	no	0.49 (± 0.07)	0.33 (± 0.06)	0.65 (± 0.11)	0.52 (± 0.13)	0.52 (± 0.03)	19.8 (± 4.5)	30 (± 10)
		10%	0.26 (± 0.01)	0.44 (± 0.01)	0.61 (± 0.01)	0.80 (± 0.01)	0.58 (± 0.01)	14.7 (± 3.6)	**10** (± 0)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	no	0.50 (± 0.08)	0.38 (± 0.03)	0.68 (± 0.03)	0.50 (± 0.13)	0.54 (± 0.03)	**19.2** (± 3.5)	24 (± 7)
		10%	0.31 (± 0.01)	0.65 (± 0.07)	0.79 (± 0.07)	0.84 (± 0.04)	0.71 (± 0.03)	15.9 (± 4.4)	**10** (± 0)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	no	**0.56** (± 0.02)	0.43 (± 0.04)	0.78 (± 0.05)	0.48 (± 0.04)	0.63 (± 0.02)	36.8 (± 6.4)	22 (± 6)
		10%	0.31 (± 0.01)	0.80 (± 0.01)	0.90 (± 0.01)	0.90 (± 0.01)	0.79 (± 0.01)	12.5 (± 3.2)	**10** (± 0)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	no	0.55 (± 0.02)	0.43 (± 0.05)	0.78 (± 0.06)	0.50 (± 0.07)	0.63 (± 0.02)	43.6 (± 12.9)	26 (± 3)
		10%	0.31 (± 0.01)	0.80 (± 0.01)	0.90 (± 0.01)	0.90 (± 0.01)	0.79 (± 0.01)	14.5 (± 4.2)	**10** (± 0)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	no	0.54 (± 0.04)	0.48 (± 0.06)	0.76 (± 0.04)	0.58 (± 0.04)	0.64 (± 0.03)	41.7 (± 7.8)	27 (± 7)
		10%	0.31 (± 0.01)	0.79 (± 0.01)	0.89 (± 0.01)	0.90 (± 0.01)	0.78 (± 0.01)	21.7 (± 2.6)	**10** (± 0)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	no	0.53 (± 0.03)	0.45 (± 0.05)	0.76 (± 0.05)	0.54 (± 0.05)	0.63 (± 0.03)	38.1 (± 8.5)	28 (± 5)
		10%	0.31 (± 0.01)	0.79 (± 0.02)	0.89 (± 0.01)	0.89 (± 0.01)	0.78 (± 0.01)	19.3 (± 4.5)	**10** (± 0)
Hu *et al.*	$L_{W_{\text{sym}}}$	no	0.54 (± 0.02)	0.43 (± 0.03)	**0.79** (± 0.04)	0.49 (± 0.05)	0.63 (± 0.02)	43.2 (± 15.5)	23 (± 3)
		10%	0.31 (± 0.02)	0.81 (± 0.01)	0.91 (± 0.01)	0.91 (± 0.01)	0.80 (± 0.01)	20.8 (± 6.3)	**10** (± 0)
	$L_{W_{\text{rw}}}$	no	0.54 (± 0.02)	0.41 (± 0.07)	0.78 (± 0.04)	0.47 (± 0.05)	0.63 (± 0.04)	47.3 (± 12.5)	24 (± 5)
		10%	0.31 (± 0.01)	0.81 (± 0.02)	0.91 (± 0.01)	0.91 (± 0.01)	0.80 (± 0.01)	22.6 (± 6.5)	**10** (± 0)
Boyd *et al.*	$L_{W_{\text{sym}}}$	no	0.52 (± 0.10)	0.45 (± 0.19)	0.72 (± 0.20)	0.69 (± 0.15)	0.68 (± 0.15)	51.3 (± 23.7)	17 (± 9)
		10%	**0.32** (± 0.01)	**0.91** (± 0.01)	**0.96** (± 0.01)	**0.96** (± 0.01)	**0.89** (± 0.01)	17.5 (± 2.5)	**10** (± 0)
	$L_{W_{\text{rw}}}$	no	0.52 (± 0.04)	**0.51** (± 0.23)	0.72 (± 0.20)	**0.76** (± 0.09)	**0.71** (± 0.12)	47.5 (± 28.2)	**15** (± 5)
		10%	**0.32** (± 0.01)	0.90 (± 0.01)	**0.96** (± 0.01)	**0.96** (± 0.01)	**0.89** (± 0.01)	20.7 (± 2.9)	**10** (± 0)

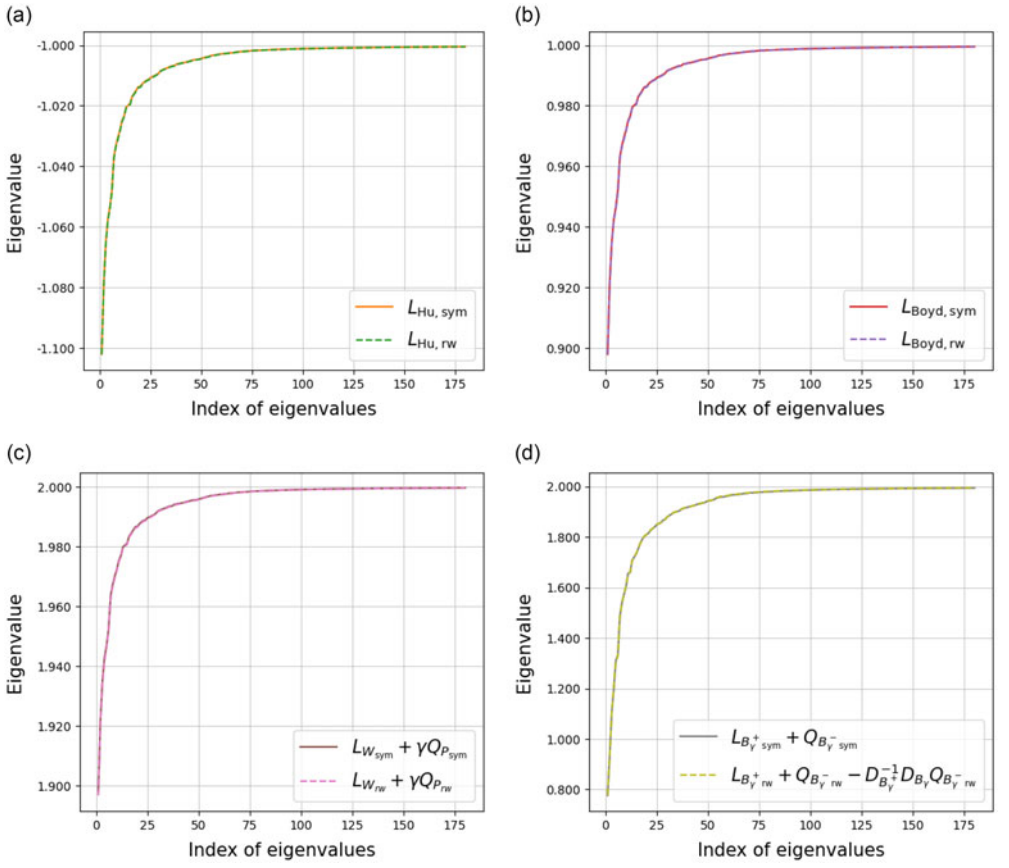


Figure 1. *MNIST: comparison of the spectra of different operators with $\gamma = 1$ under the NG null model. In each of the plots, one of the two curves is hidden behind the other one.*

results shown in section 4.2.2. of [33]. Therefore, choosing a value for η in the partition-based criterion (48) larger than $= 10^{-5}$ (which is the value used to produce the results in Table 3) could lead to similar modularity scores but fewer iterations.

The three rightmost columns show (from left to right) the average time (over the 20 runs) that the computation of the eigenvalues and eigenvectors takes, the average total time it takes to perform all the iterations in the respective MBO scheme and the average number of iterations of the MBO scheme that are required. All times presented in these tables (and those that follow later in the paper) are given in seconds. It can be seen from Tables 3 and 4 that for all methods, except MMBO Algorithm 1 with the modularity-based stopping criterion, the time for computing the eigenvalues and eigenvectors is smaller than the time for performing the iterations, yet still contributes non-trivially to the run time. The number of iteration steps is considerably smaller if the modularity-based stopping criterion is used than if the partition-based stopping criterion is used. Within each table, thus with fixed stopping condition, the necessary average number of iterations is similar for all methods. By far the shortest average time per iteration (which can be obtained by dividing the average time for the MBO iterations by the average number of iterations) among the methods considered in Tables 3 and 4 is needed for MMBO Algorithm 1.

Results obtained with the different methods under the NG null model are presented in Tables 5, 6 and 7. The modularity scores (with the NG null model) and the various metrics from Section 7.1.3 are given, where the values are averages over 20 runs, followed by the maximum deviation from the average in parentheses. The total average run time (including both the time needed for computing the eigenvalues

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

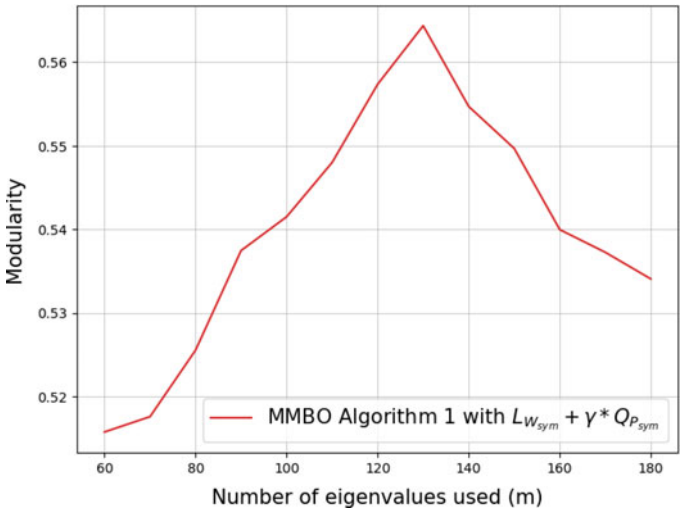


Figure 2. *MNIST: relationship between the number of eigenvalues used and modularity. The MMBO Algorithm 1 uses the modularity-based stopping condition (49) and* $\gamma = 1$.

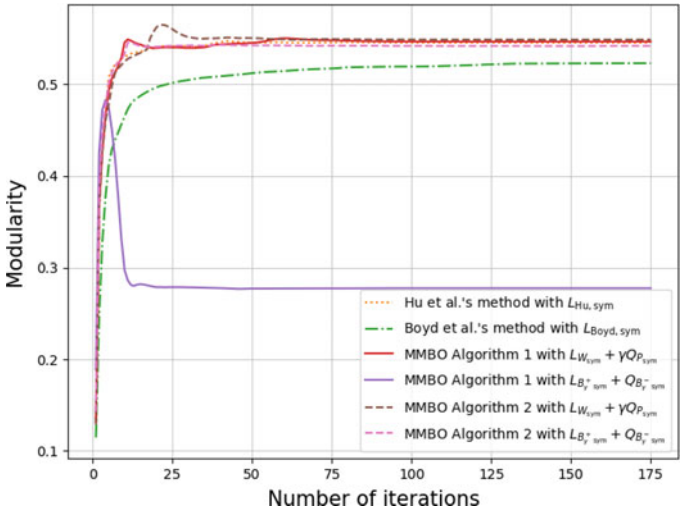


Figure 3. *MNIST: Modularity score versus number of iterations, obtained with* $\gamma = 1$ *without stopping criterion.*

and eigenvectors as well as the MBO iterations) of each algorithm (and its maximum deviation) is given in seconds in the second-rightmost column. The rightmost column lists the average number of non-empty clusters (and its maximum deviation) that is returned by each method. As in Tables 3 and 4, the initial condition U^0 may change with each run. However, similar to the previous results, such changes in the initial conditions do not significantly influence the modularity scores or scores of other classification metrics obtained by the algorithm.

Many of the methods obtain modularity scores that are larger than 0.52, with the Louvain method giving the highest value (0.58). Comparing the results presented in Tables 6 and 7, it can be seen that the stopping criterion might have a certain (small) impact on the modularity score. Sometimes, even higher

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

modularity scores are obtained with the modularity-based stopping criterion than with the partition-based stopping criterion, which indicates that in these cases the modularity score decreases during the further iterations if the partition-based stopping criterion is utilised; this behaviour is illustrated in Figure 3. It is noteworthy that the best MMBO scheme generally provides higher modularity scores than the methods of Boyd *et al.* The fastest method is MMBO Algorithm 1. In particular, both MMBO Algorithm 1 and Algorithm 2 are significantly faster than the Louvain algorithm, with a smaller loss in modularity (average). In line with what we observed earlier, also this algorithm is even faster if the modularity-based stopping criterion is used (Table 7) than if the partition-based stopping criterion is used (Table 6). The high modularity score obtained by the Louvain method, however, leads to the conclusion that the value $\gamma = 1$ which we used here is not optimal if the goal were to have high modularity scores correspond to partitions that are close to the ground truth. Because that is only of secondary concern in this paper, we will not vary the value of γ in this example. It is also worth noting that, although the modularity obtained by the Leiden algorithm is not as high as that reached by the Louvain method, the former method achieves significantly better results in some of the other quantities of interest. Additionally, its run time is shorter (much shorter in some cases) than that of all other methods.

Finally for this example, we study the impact of a very mild form of semi-supervision on the modularity scores and the other quantities of interest. To this end, we uniformly chose $70,000 \times 10\% = 7,000$ nodes at random and assigned them true community assignments obtained from the available ground truth (GT). We only do this in the initial condition of the algorithm. It is possible to incorporate stronger fidelity to this partial ground truth in the MBO schemes by introducing an extra fidelity-forcing term into the linear-dynamics step, along the lines of what was done in Budd *et al.* [12]. Since our primary focus in this paper is modularity optimisation and fidelity to a ground truth is only of secondary concern, we have not pursued that option here, but it can be an interesting direction for future research.

The obtained results are depicted in Table 7. It can be observed that, on the one hand, the modularity scores of the MBO-based methods are noticeably lower. This suggests that the initial condition U^0 of the algorithm influences the outcome. But, on the other hand, the other classification metrics increase significantly for all approaches when using 10% of ground truth. In addition, using supervision leads to a notable reduction of computing times (number of iterations) under the modularity-based stopping criterion.

7.3. Stochastic block model

The SBM [32] evolved from the study of social networks. The SBM is a model to create random graphs in which the node set is split into separate groups, also known as blocks, that determine the edge connectivity structure. The SBM is constructed by generating an undirected edge between each pair of nodes independently. The probability of an edge linking two nodes is solely determined by the blocks to which the nodes belong.

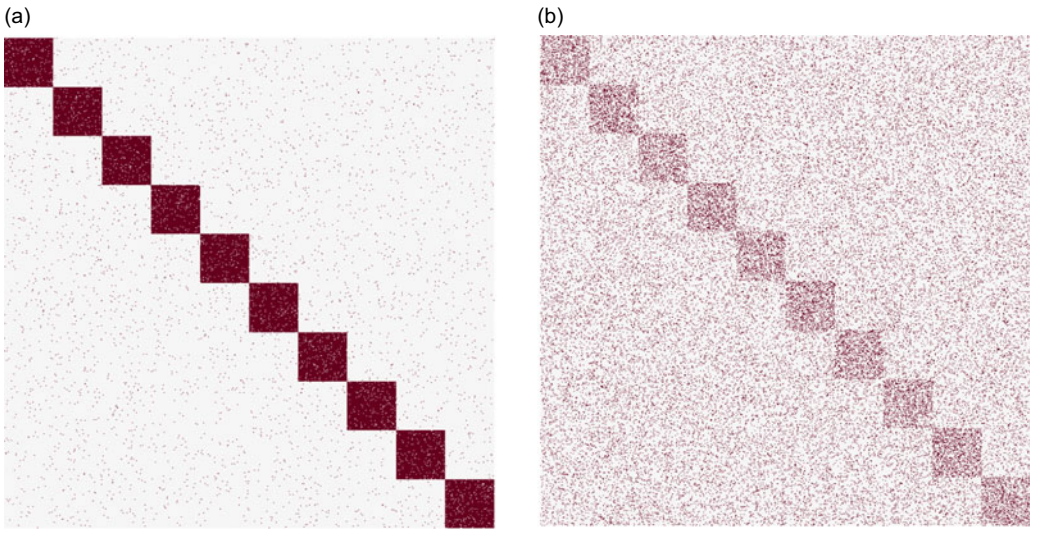
We start the unweighted SBM graph construction with a node set V that is partitioned into several equally-sized subsets called blocks. In a general undirected SBM setting (without self-loops), undirected edges are constructed between each pair of distinct nodes independently with a probability that depends only on the block memberships of both nodes. We restrict ourselves to a specific setting in which the probability for each (non-self-loop) intra-block connection, p_{same}, is the same, and the probability for each inter-block connection, p_{diff}, is the same. We want p_{diff} to be smaller than p_{same} and study two types of SBMs: strong and weak community structures. To obtain a strong community structure, we set $p_{\text{same}} = 0.95$, and $p_{\text{diff}} = 0.01$. In contrast, in the weak community structure, the probabilities are $p_{\text{same}} = 0.3$ and $p_{\text{diff}} = 0.1$. Table 8 summarises the parameters we use to construct realisations of the SBM. Examples of adjacency matrices of realisations of the strong and weak community structure at $K = 10$ are shown in Figure 4, where the dark colour indicates the existence of an edge and white the absence.

By construction, the ground truth community assignments of each node in a realisation of an SBM are known.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 8. *Parameter settings used to construct the SBM*

Parameter	Value
Nodes	$\lvert V\rvert = 3000$
# blocks	10
Block size	$\lvert V\rvert$/ Number of blocks
Probability	Strong community structure: $p_{\text{same}} = 0.95$, $p_{\text{diff}} = 0.01$
	Weak community structure: $p_{\text{same}} = 0.3$, $p_{\text{diff}} = 0.1$



SBM with the strong community structure.

SBM with the weak community structure.

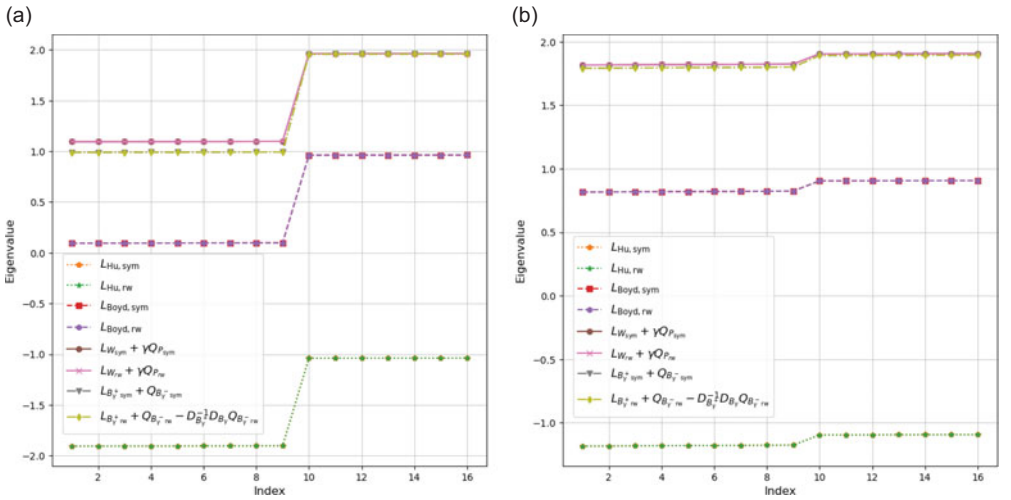
Figure 4. *SBM: Adjacency matrices of realisations of the strong and weak community structure where the number of blocks is* 10.

Figure 5 depicts the first 16 eigenvalues of L_{Hu}, L_{Boyd} (using $L_{W_{\text{sym}}}$ or $L_{W_{\text{rw}}}$ as graph Laplacian) and $L_{\text{mix}} \in \{L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}, L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}, L_{B^+_{\gamma}{}_{\text{sym}}} + Q_{B^-_{\gamma}{}_{\text{sym}}}, L_{B^+_{\gamma}{}_{\text{rw}}} + Q_{B^-_{\gamma}{}_{\text{rw}}} - D^{-1}_{B^+_{\gamma}} D_{B_{\gamma}} Q_{B^-_{\gamma}{}_{\text{rw}}}\}$ for a realisation of an SBM with 10-block strong and weak community structures, $\gamma = 1$, and the NG null model. It is worth noting that there is a sudden jump in the eigenvalues for both the strong community structure (Figure 5a) and weak community structure (Figure 5b) at the tenth eigenvalue.[60] The graph on which Figure 5 is based did not require the use of the Nyström extension with QR decomposition (Section 6.4) for the eigenvalue and eigenvector computation. We have used the Nyström extension only for graphs with more than 10,000 nodes. In this case, we use the implicitly restarted Lanczos method [44] to find the eigenvalues and eigenvectors.

The parameters used for performing the MMBO schemes, Hu *et al.*'s method and Boyd *et al.*'s method are given in Table 9. To motivate the choice of m, the number of eigenvalues we used in different methods, we investigate the effect of the number of used eigenvectors on the final result. In Figure 6, we plot the modularity score produced by varying m, using the eigenvalues from Figure 5 which are obtained for a single realisation of an SBM with 10 blocks. For each value of m, each algorithm is run 20 times and the average modularity scores are plotted. To avoid cluttering the plots, we have only included results for the algorithms that use the symmetrically normalised (signless) Laplacians. Similar

[60] Remark D.4 in Appendix D discusses possible relations of the occurrences of jumps of the eigenvalues of $L_{\text{mix}} = L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$ (or $L_{\text{mix}} = L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$) on the one hand and $L_{W_{\text{sym}}}$ (or $L_{W_{\text{rw}}}$) on the other hand, for $P = P^{\text{NG}}$, the NG null model.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press



SBM with the strong community structure. SBM with the weak community structure.

Figure 5. *SBM with strong and weak community structure: spectra of $L_{Hu,sym}$, $L_{Hu,rw}$, $L_{Boyd,sym}$, $L_{Boyd,rw}$ and four choices of $L_{mix} \in \{L_{W_{sym}} + \gamma Q_{P_{sym}}, L_{W_{rw}} + \gamma Q_{P_{rw}}, L_{B^+_{\gamma\ sym}} + Q_{B^-_{\gamma\ sym}}, L_{B^+_{\gamma\ rw}} + Q_{B^-_{\gamma\ rw}} - D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\ rw}}\}$ with $\gamma = 1$ and the NG null model, for a single realisation of an SBM with 10 blocks. The following graphs overlap: $L_{Hu,sym}$ and $L_{Hu,rw}$; $L_{Boyd,sym}$ and $L_{Boyd,rw}$; $L_{W_{sym}} + \gamma Q_{P_{sym}}$ and $L_{W_{rw}} + \gamma Q_{P_{rw}}$ (which is expected thanks to Remark 5.2); $L_{B^+_{1\ sym}} + Q_{B^-_{1\ sym}}$ and (using that $D_{B_1} = 0$ by (20)) $L_{B^+_{1\ rw}} + Q_{B^-_{1\ rw}} - D^{-1}_{B^+_1} D_{B_1} Q_{B^-_{1\ rw}} = L_{B^+_{1\ rw}} + Q_{B^-_{1\ rw}}$ (which is expected from Remark 5.3).*

results were obtained when using the random walk (signless) Laplacians. Upon examining Figure 6, it is apparent that all depicted methods achieve the highest modularity score (within the depicted domains for m) at $m = 12$ for the strong community structure (as shown in Figure 6a) and at $m = 10$ for the weak community structure (illustrated in Figure 6b). Moreover, the maximum modularity scores achieved by the methods are approximately the same (see also Tables 11 and 12). The combined observations from Figures 5 and 6 suggest that the number of eigenvectors chosen should be at least the same as the number of clusters, that is, $m \geq K$.

Tables 10–12 present results that are obtained using the same realisation of an SBM with 10 blocks that was used for Figures 5 and 6. Each method was run 20 times, and the average modularity scores and other quantities of interest are reported in the tables, followed by the maximum deviations from the average in parentheses. The reported times are average time per run, including the computation of eigenvectors and eigenvalues for the methods that require that. The number of iterations refers specifically to the iterations of the respective MBO schemes in both MMBO algorithms, Hu *et al.*'s method and Boyd *et al.*'s method. For those four methods, stopping criterion (48) was used. For the methods that require a prescribed value of K, we use K found by the Leiden algorithm, that is, $K = 10$ for both the strong community structure and the weak community structure, since the highest modularity scores in Table 10 are obtained at this value. Using what we learned from Figure 6, for both MMBO algorithms and for Hu's method and Boyd's method we choose $m = 12$ for the SBM realisation with strong community structure and $m = 10$ for the one with weak community structure.

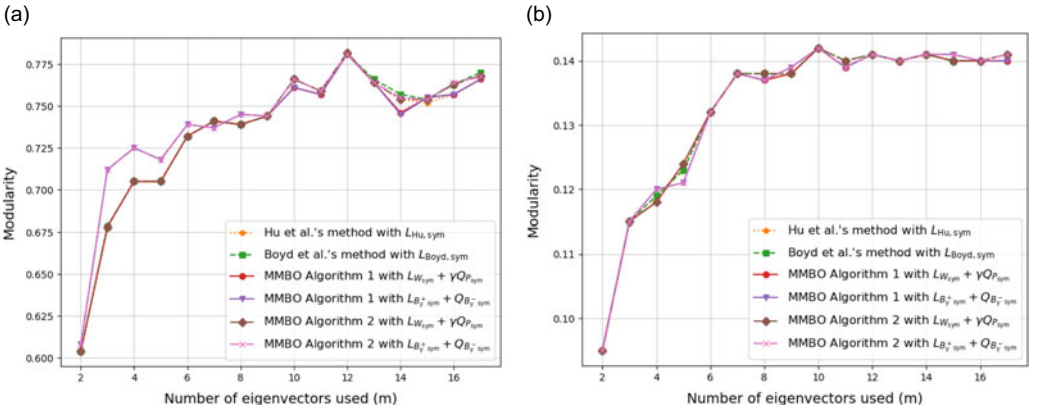
We recall the variability of the initial condition U^0 in Algorithms 1 and 2, where each node is randomly allocated to a community under the constraint that each community contains at least one node. This initial assignment can differ in each iteration. However, such variations in the initial conditions have a negligible effect on the modularity score that the algorithms yield.

Consider first the results for the SBM with strong community structure, as presented in Tables 10 and 11. It can be observed that the best modularity score, with the value 0.813, is computed with the

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 9. *Parameter setting of the MMBO schemes, Hu et al.'s and Boyd et al.'s methods in SBM*

Parameter	Value
Clusters	*K*equal to # blocks
N_t	3
η	10^{-4}
γ	1



Different choices of m for SBM with strong community structure.

Different choices of m for SBM with weak community structure.

Figure 6. *SBM with strong and weak community structures: modularity depending on the number of eigenvalues used (m) for SBM blocks are* 10*. The number of clusters K used by the MMBO schemes, Hu et al.'s and Boyd et al.'s methods are obtained from Leiden algorithm, that is,* $K = 10$ *for both the strong community structure and the weak community structure. All methods use* $\gamma = 1$*, the partitioned-based stopping condition (48) and the NG null model. The red circle solid curve and purple triangle solid curve are overlapped by the brown diamond dashed curve and pink octagon dashed curve, respectively.*

Louvain method, Leiden algorithm, CNM and spectral clustering. Also all other evaluation measures (except the run times) are identical for these methods; in fact, these four methods return exactly the ground truth. The Leiden algorithm is the fastest of all methods and is one of the methods that gives the highest modularity score. Although the Louvain method and spectral clustering also give high modularity scores, these methods are slower than the MBO-based methods (i.e., both MMBO algorithms, Hu's method and Boyd's method) in Table 11. The average modularity scores for both MMBO algorithms, which are around 0.77, are similar to Hu *et al.*'s and Boyd *et al.*'s methods, but not decisively so when taking into account the reported maximum deviations from the average. The modularity scores, ARI, (inverse) purity and NMI in Table 11 are also very similar for these MBO-based methods.

The results for the SBM with weak community structure are presented in Tables 10 and 12. Their evaluation leads to very similar conclusions as for the case of the strong community structure. Noticeable differences are that the Louvain, CNM and spectral clustering methods do no longer reproduce the ground truth. The Leiden algorithm not only gives the highest modularity and highest value for all other evaluation metrics but also has the shortest run time of all methods.

A surprising result from Tables 10, 11 and 12 is that in the strong community structure case, none of the MBO-based methods return the same number of non-empty clusters as the ground truth has, even though the Louvain method, the Leiden algorithm, the CNM method and spectral clustering all do

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 10. *SBM: average NG modularity, other classification metrics scores, and average computation time per run obtained from* 20 *runs. The best average results for the strong and for the weak community structure in each column are shown in boldface. For the number of non-empty clusters we consider the one closest to the ground truth number* 10 *to be 'best' in this context*

Community structure	Method	NG modularity	ARI	Purity	Inv. purity	NMI	Non-empty clusters	Time (s)
Strong	CNM	**0.813** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**10** ($\pm$ 0)	41.1 ($\pm$ 2.6)
	Louvain	**0.813** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**10** ($\pm$ 0)	4.6 ($\pm$ 1.5)
	Leiden	**0.813** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**10** ($\pm$ 0)	**0.6** ($\pm$ 0.2)
Weak	CNM	0.107 ($\pm$ 0.0)	0.36 ($\pm$ 0.0)	0.39 ($\pm$ 0.0)	0.92 ($\pm$ 0.0)	0.63 ($\pm$ 0.0)	4 ($\pm$ 0)	61.9 ($\pm$ 1.7)
	Louvain	0.145 ($\pm$ 0.001)	0.88 ($\pm$ 0.01)	0.88 ($\pm$ 0.01)	**1.0** ($\pm$ 0.0)	0.96 ($\pm$ 0.01)	9 ($\pm$ 1)	7.1 ($\pm$ 1.4)
	Leiden	**0.149** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**1.0** ($\pm$ 0.0)	**10** ($\pm$ 0)	**1.0** ($\pm$ 0.4)

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 11. *SBM with strong community structure: average performance of algorithms regarding modularity scores, various classification indicators, average time per run, and average number of iterations per run. The number of clusters K used by spectral clustering, MMBO schemes, Hu et al.'s, and Boyd et al.'s methods are obtained from the Leiden algorithm, that is, $K = 10$. Moreover, for the MMBO schemes, Hu et al.'s method and Boyd et al.'s method, we choose $m = 12$. The best average results in each column are shown in boldface (we exclude the ground truth numbers). For the number of non-empty clusters we consider the one closest to the ground truth number to be 'best' in this context*

Method		NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters	# iter.
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.779 ($\pm$0.032)	0.79 ($\pm$0.08)	0.77 ($\pm$0.13)	**1.0** ($\pm$0.0)	0.93 ($\pm$0.06)	**1.3** ($\pm$0.3)	8 ($\pm$2)	4 ($\pm$2)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.762 ($\pm$0.025)	0.75 ($\pm$0.04)	0.73 ($\pm$0.03)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.02)	**1.3** ($\pm$0.4)	8 ($\pm$1)	3.2 ($\pm$0.8)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.779 ($\pm$0.035)	0.79 ($\pm$0.08)	0.77 ($\pm$0.13)	**1.0** ($\pm$0.0)	0.93 ($\pm$0.06)	**1.3** ($\pm$0.5)	8 ($\pm$2)	4 ($\pm$1)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.771 ($\pm$0.025)	0.76 ($\pm$0.04)	0.76 ($\pm$0.03)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.02)	1.4 ($\pm$0.4)	8 ($\pm$1)	**3.0** ($\pm$1.0)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.779 ($\pm$0.021)	0.81 ($\pm$0.08)	0.80 ($\pm$0.10)	**1.0** ($\pm$0.0)	0.94 ($\pm$0.05)	**1.3** ($\pm$0.4)	8 ($\pm$2)	4 ($\pm$2)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.763 ($\pm$0.024)	0.76 ($\pm$0.05)	0.73 ($\pm$0.05)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.02)	1.5 ($\pm$0.6)	8 ($\pm$1)	3.8 ($\pm$0.2)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.779 ($\pm$0.022)	0.81 ($\pm$0.08)	0.80 ($\pm$0.10)	**1.0** ($\pm$0.0)	0.94 ($\pm$0.05)	**1.3** ($\pm$0.5)	8 ($\pm$2)	4 ($\pm$2)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.771 ($\pm$0.020)	0.78 ($\pm$0.07)	0.77 ($\pm$0.06)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.03)	1.5 ($\pm$0.4)	8 ($\pm$1)	3.6 ($\pm$0.4)
Hu *et al.*	$L_{W_{\text{sym}}}$	0.779 ($\pm$0.018)	0.80 ($\pm$0.10)	0.80 ($\pm$0.12)	**1.0** ($\pm$0.0)	0.93 ($\pm$0.04)	2.0 ($\pm$0.4)	8 ($\pm$1)	5 ($\pm$2)
	$L_{W_{\text{rw}}}$	0.770 ($\pm$0.020)	0.79 ($\pm$0.10)	0.78 ($\pm$0.11)	**1.0** ($\pm$0.0)	0.93 ($\pm$0.04)	1.9 ($\pm$0.4)	8 ($\pm$1)	5 ($\pm$1)
Boyd *et al.*	$L_{W_{\text{sym}}}$	0.779 ($\pm$0.024)	0.81 ($\pm$0.09)	0.80 ($\pm$0.10)	**1.0** ($\pm$0.0)	0.93 ($\pm$0.04)	1.6 ($\pm$0.5)	8 ($\pm$1)	4.7 ($\pm$0.1)
	$L_{W_{\text{rw}}}$	0.767 ($\pm$0.023)	0.80 ($\pm$0.09)	0.79 ($\pm$0.10)	**1.0** ($\pm$0.0)	0.92 ($\pm$0.02)	1.5 ($\pm$0.5)	8 ($\pm$1)	4.3 ($\pm$0.4)
Spectral clustering		**0.813** ($\pm$0.0)	**1.0** ($\pm$0.0)	**1.0** ($\pm$0.0)	**1.0** ($\pm$0.0)	**1.0** ($\pm$0.0)	2.4 ($\pm$0.4)	**10** ($\pm$0)	–
Ground truth		0.813	1.0	1.0	1.0	1.0	–	10	–

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 12. *SBM with weak community structure: average performance of algorithms regarding modularity scores, various classification indicators, average time per run, and average number of iterations per run. The number of clusters K used by spectral clustering, MMBO schemes, Hu et al.'s, and Boyd et al.'s methods are obtained from the Leiden algorithm, that is, $K = 10$. Moreover, for the MMBO schemes, Hu et al.'s method and Boyd et al.'s method, we choose $m = 10$. The best average results in each column are shown in boldface (we exclude the ground truth numbers). For the number of non-empty clusters we consider the one closest to the ground truth number to be 'best' in this context*

Method		NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters	# iter.
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.141 ($\pm$0.008)	0.77 ($\pm$0.10)	0.75 ($\pm$0.15)	**1.0** ($\pm$0.0)	0.92 ($\pm$0.06)	1.5 ($\pm$0.6)	8 ($\pm$2)	**5** ($\pm$2)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.140 ($\pm$0.003)	0.76 ($\pm$0.08)	0.74 ($\pm$0.10)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.03)	1.8 ($\pm$0.5)	7 ($\pm$1)	**5** ($\pm$1)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.141 ($\pm$0.007)	0.77 ($\pm$0.10)	0.75 ($\pm$0.15)	**1.0** ($\pm$0.0)	0.92 ($\pm$0.06)	**1.4** ($\pm$0.4)	8 ($\pm$2)	**5** ($\pm$3)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.140 ($\pm$0.003)	0.76 ($\pm$0.08)	0.74 ($\pm$0.10)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.03)	1.6 ($\pm$0.4)	7 ($\pm$1)	**5** ($\pm$1)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.141 ($\pm$0.008)	0.77 ($\pm$0.10)	0.75 ($\pm$0.15)	**1.0** ($\pm$0.0)	0.92 ($\pm$0.06)	1.7 ($\pm$0.5)	8 ($\pm$2)	**5** ($\pm$3)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.140 ($\pm$0.003)	0.76 ($\pm$0.07)	0.74 ($\pm$0.09)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.03)	2.2 ($\pm$0.5)	7 ($\pm$1)	**5** ($\pm$1)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.141 ($\pm$0.007)	0.77 ($\pm$0.10)	0.74 ($\pm$0.14)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.06)	1.6 ($\pm$0.6)	7 ($\pm$2)	6 ($\pm$1)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.140 ($\pm$0.003)	0.76 ($\pm$0.07)	0.74 ($\pm$0.09)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.03)	2.0 ($\pm$0.7)	7 ($\pm$1)	**5** ($\pm$1)
Hu *et al.*	$L_{W_{\text{sym}}}$	0.141 ($\pm$0.004)	0.75 ($\pm$0.10)	0.73 ($\pm$0.13)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.05)	2.3 ($\pm$0.3)	7 ($\pm$1)	**5** ($\pm$1)
	$L_{W_{\text{rw}}}$	0.141 ($\pm$0.006)	0.76 ($\pm$0.13)	0.74 ($\pm$0.11)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.06)	2.3 ($\pm$0.3)	7 ($\pm$2)	**5** ($\pm$2)
Boyd *et al.*	$L_{W_{\text{sym}}}$	0.141 ($\pm$0.005)	0.77 ($\pm$0.10)	0.75 ($\pm$0.15)	**1.0** ($\pm$0.0)	0.92 ($\pm$0.06)	1.9 ($\pm$0.4)	8 ($\pm$2)	7 ($\pm$3)
	$L_{W_{\text{rw}}}$	0.140 ($\pm$0.005)	0.76 ($\pm$0.10)	0.74 ($\pm$0.14)	**1.0** ($\pm$0.0)	0.91 ($\pm$0.05)	1.8 ($\pm$0.4)	7 ($\pm$2)	8 ($\pm$3)
Spectral clustering		**0.146** ($\pm$0.001)	**0.90** ($\pm$0.01)	**0.90** ($\pm$0.0)	**1.0** ($\pm$0.0)	**0.96** ($\pm$0.01)	3.0 ($\pm$0.7)	**9** ($\pm$1)	–
Ground truth		0.149	1.0	1.0	1.0	1.0	–	10	–

Table 13. *Two cows: parameter settings for the Nyström extension and edge weights in (59) (left) and parameter setting of the MMBO schemes (right)*

Parameter	Value	Parameter	Value
k	500	Clusters	Determined by Louvain's method or $K = 3$
σ	50	m	$m = K$
		N_t	3
		η	10^{-5}
		γ	1

while also achieving a higher modularity score. This observation remains true even in the case with weak community structure, except for the CNM method, which performs poorly. In the MNIST example, the number of non-empty clusters found by the MBO-based methods (Tables 6 and 7) is a lot closer to that of the ground truth than to the number found by the Louvain method and Leiden algorithm (Table 5), even though all these methods give similar average modularity scores (with the Louvain method even giving the highest average score). Looking ahead to the 'two cows' example of Section 7.4, we can draw a similar conclusion as in the MNIST case when we compare the number of clusters found by the MBO-based methods (Tables 15 and 16) with the Louvain method and Leiden algorithm (Table 14); in that case also, the CNM method performs quite well in this regard. Despite that, the modularity scores that are obtained by the Leiden algorithm and, especially, the Louvain method, in that setting are noticeably higher than those obtained by the MBO-based methods. In this context, it is useful to note that Hu's method and Boyd's method were not tested on SBMs in [33] and [7], so we have no other tests on SBMs for the MBO-based modularity optimisation methods with which to compare the results that we obtained here.

7.4. Two cows

The 'two cows' image is a 213×320 RGB image [4, 52]. The classification task consists in identifying pixels that include comparable components, such as sky, cows and grass.

We model the image by a graph as follows. With each pixel i of the image, we associated a node i in the graph; hence, $|V| = 213 \times 320 = 68,160$. We then define the weighted adjacency matrix W by assigning to each node $i \in V$ a *feature vector* $x_i \in \mathbb{R}^{27}$ and defining ω_{ij} to be as in (59) with $\sigma = 50$. The feature vector of node i contains the three RGB intensity levels for each of the nine pixels in the three-by-three patch centred at pixel i in the image. To obtain full patches for pixels at the edge of the image, we use symmetric padding, which consists of adding an extra row or column of pixels around the edge of the image that mirror the pixel values that are at the edge of the image.

Actually constructing the matrix $W \in \mathbb{R}^{68,160 \times 68,160}$ would require too much memory and computing time to be feasible, if even possible. Due to the Nyström extension with QR decomposition from Section 6.4, however, we can still compute the eigenvalues and eigenvectors of L_{mix} for the MMBO schemes or of the method-specific matrices for Hu's method and Boyd's method. All parameter values that we use are listed in Table 13.

Observing the original RGB image, we manually cluster the pixels into three 'communities' representing the sky, the grass and both cows. In the 'ground truth' image in Figure 7, we have represented each cluster by a different colour.

Figure 7 and 8 display the image segmentation outcomes for the various methods. We have run each method 20 times and display the result with the highest modularity score. In Algorithms 1 and 2, the initial condition U^0 varies, as each node is randomly assigned to a community with the constraint that each community contains at least one node. Although this assignment may differ in each iteration, these variations have a negligible impact on the resulting modularity score. Tables 14–16 show the average modularity scores and other quantities of interest, followed by the maximum deviation (in absolute value)

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 14. *Two cows: average performance of algorithms regarding modularity scores, various classification metrics, and computation time per run under NG null model. The best average result in each column is shown in boldface (we exclude the ground truth numbers). For the number of non-empty clusters we consider the one closest to the ground truth number* 3 *to be 'best' in this context*

Method	NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters
Louvain	**0.92** (± 0.01)	0.017 (± 0.001)	0.98 (± 0.01)	0.05 (± 0.0)	0.30 (± 0.01)	50.8 (± 3.3)	168 (± 10)
CNM	0.70 (± 0.02)	**0.31** (± 0.01)	0.75 (± 0.01)	**0.66** (± 0.0)	0.45 (± 0.01)	1586.5 (± 183.4)	**7** (± 1)
Leiden	0.87 (± 0.01)	0.23 (± 0.03)	**0.99** (± 0.01)	0.31 (± 0.04)	**0.52** (± 0.02)	**3.8** (± 0.5)	16 (± 2)
Ground truth	0.53	1.0	1.0	1.0	1.0	–	3

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 15. *Two cows: average performance of algorithms under the NG null model regarding modularity scores, various classification metrics, and computation time per run under the NG model. In all cases, $K = 168$ is applied to spectral clustering, MMBO schemes, Hu et al.'s method, and Boyd et al.'s method. Note that for the MMBO schemes, Hu et al.'s and Boyd et al.'s methods, we choose $m = K = 168$ and use modularity-based stopping condition (49). The best average results in each column are shown in boldface. For the number of non-empty clusters we consider the one closest to the ground truth number 3 to be 'best' in this context*

Method		NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters	iter. iter.
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.79 ($\pm$0.06)	0.45 ($\pm$0.08)	**0.98** ($\pm$0.01)	0.64 ($\pm$0.08)	**0.66** ($\pm$0.07)	15.8 ($\pm$3.3)	7 ($\pm$1)	34 ($\pm$16)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.77 ($\pm$0.04)	0.42 ($\pm$0.03)	**0.98** ($\pm$0.01)	0.58 ($\pm$0.06)	0.64 ($\pm$0.02)	**13.1** ($\pm$2.5)	7 ($\pm$2)	37 ($\pm$10)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.55 ($\pm$0.08)	**0.48** ($\pm$0.09)	0.80 ($\pm$0.02)	0.75 ($\pm$0.05)	0.45 ($\pm$0.06)	20.4 ($\pm$8.9)	8 ($\pm$2)	59 ($\pm$20)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.56 ($\pm$0.08)	0.44 ($\pm$0.07)	0.78 ($\pm$0.03)	**0.83** ($\pm$0.02)	0.46 ($\pm$0.02)	22.5 ($\pm$3.9)	**6** ($\pm$2)	41 ($\pm$17)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	0.79 ($\pm$0.05)	0.43 ($\pm$0.07)	**0.98** ($\pm$0.01)	0.64 ($\pm$0.05)	0.64 ($\pm$0.05)	20.7 ($\pm$5.1)	8 ($\pm$1)	37 ($\pm$20)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	0.78 ($\pm$0.03)	0.42 ($\pm$0.02)	**0.98** ($\pm$0.01)	0.59 ($\pm$0.05)	0.65 ($\pm$0.05)	23.5 ($\pm$5.8)	**6** ($\pm$2)	33 ($\pm$16)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.58 ($\pm$0.15)	0.37 ($\pm$0.08)	0.79 ($\pm$0.04)	0.69 ($\pm$0.12)	0.43 ($\pm$0.07)	30.2 ($\pm$11.6)	8 ($\pm$2)	45 ($\pm$17)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.60 ($\pm$0.12)	0.35 ($\pm$0.07)	0.80 ($\pm$0.04)	0.70 ($\pm$0.10)	0.43 ($\pm$0.05)	26.3 ($\pm$7.5)	12 ($\pm$4)	**29** ($\pm$13)
Hu *et al.*	$L_{W_{\text{sym}}}$	0.76 ($\pm$0.04)	0.41 ($\pm$0.02)	**0.98** ($\pm$0.01)	0.57 ($\pm$0.02)	0.64 ($\pm$0.01)	42.8 ($\pm$15.0)	8 ($\pm$1)	44 ($\pm$13)
	$L_{W_{\text{rw}}}$	0.78 ($\pm$0.04)	0.34 ($\pm$0.03)	**0.98** ($\pm$0.01)	0.49 ($\pm$0.06)	0.58 ($\pm$0.03)	44.5 ($\pm$12.7)	13 ($\pm$3)	44 ($\pm$28)
Boyd *et al.*	$L_{W_{\text{sym}}}$	0.76 ($\pm$0.03)	0.41 ($\pm$0.02)	**0.98** ($\pm$0.02)	0.57 ($\pm$0.03)	0.64 ($\pm$0.02)	34.9 ($\pm$10.6)	8 ($\pm$2)	40 ($\pm$32)
	$L_{W_{\text{rw}}}$	0.78 ($\pm$0.03)	0.41 ($\pm$0.02)	**0.98** ($\pm$0.02)	0.64 ($\pm$0.03)	0.64 ($\pm$0.03)	32.4 ($\pm$13.1)	13 ($\pm$2)	39 ($\pm$26)
Spectral clustering		**0.85** ($\pm$0.02)	0.03 ($\pm$0.0)	0.96 ($\pm$0.01)	0.10 ($\pm$0.01)	0.27 ($\pm$0.01)	52.7 ($\pm$5.1)	168 ($\pm$0)	–

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Table 16. *Two cows: average performance of algorithms regarding modularity scores, various classification metrics, and computation time per run under the NG model. In all cases, $K = 3$ is applied to spectral clustering, MMBO schemes, Hu et al.'s method, and Boyd et al.'s method. Note that for the MMBO schemes, Hu et al.'s and Boyd et al.'s methods, we choose $m = K = 3$ and use modularity-based stopping condition (49). The best average results in each column are shown in boldface. For the number of non-empty clusters we consider the one closest to the ground truth number 3 to be 'best' in this context*

Method		NG modularity	ARI	Purity	Inv. purity	NMI	Time (s)	Non-empty clusters	# iter.
MMBO Algorithm 1	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	**0.54** ($\pm$0.08)	**0.95** ($\pm$0.06)	**0.99** ($\pm$0.05)	**0.98** ($\pm$0.04)	**0.92** ($\pm$0.10)	5.1 ($\pm$1.3)	**3** ($\pm$1)	**7** ($\pm$5)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	**0.54** ($\pm$0.06)	0.94 ($\pm$0.06)	**0.99** ($\pm$0.04)	**0.98** ($\pm$0.03)	**0.92** ($\pm$0.10)	**4.5** ($\pm$1.0)	**3** ($\pm$1)	8 ($\pm$5)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.50 ($\pm$0.03)	0.88 ($\pm$0.29)	0.96 ($\pm$0.17)	**0.98** ($\pm$0.03)	0.85 ($\pm$0.31)	7.5 ($\pm$2.2)	**3** ($\pm$1)	11 ($\pm$8)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.52 ($\pm$0.03)	0.85 ($\pm$0.18)	0.93 ($\pm$0.11)	0.96 ($\pm$0.02)	0.84 ($\pm$0.27)	6.2 ($\pm$1.7)	**3** ($\pm$1)	10 ($\pm$5)
MMBO Algorithm 2	$L_{W_{\text{sym}}}, Q_{P_{\text{sym}}}$	**0.54** ($\pm$0.08)	**0.95** ($\pm$0.06)	**0.99** ($\pm$0.04)	**0.98** ($\pm$0.04)	**0.92** ($\pm$0.09)	6.3 ($\pm$1.5)	**3** ($\pm$1)	**7** ($\pm$4)
	$L_{W_{\text{rw}}}, Q_{P_{\text{rw}}}$	**0.54** ($\pm$0.06)	0.94 ($\pm$0.08)	0.97 ($\pm$0.05)	**0.98** ($\pm$0.05)	**0.92** ($\pm$0.13)	5.5 ($\pm$1.9)	**3** ($\pm$1)	8 ($\pm$6)
	$L_{B^+_{\text{sym}}}, Q_{B^-_{\text{sym}}}$	0.50 ($\pm$0.04)	0.88 ($\pm$0.23)	0.96 ($\pm$0.17)	**0.98** ($\pm$0.03)	0.85 ($\pm$0.21)	9.7 ($\pm$2.6)	**3** ($\pm$1)	13 ($\pm$7)
	$L_{B^+_{\text{rw}}}, Q_{B^-_{\text{rw}}}$	0.52 ($\pm$0.03)	0.90 ($\pm$0.20)	0.96 ($\pm$0.13)	0.97 ($\pm$0.02)	0.84 ($\pm$0.17)	7.1 ($\pm$1.6)	**3** ($\pm$1)	10 ($\pm$4)
Hu *et al.*	$L_{W_{\text{sym}}}$	**0.54** ($\pm$0.05)	0.90 ($\pm$0.07)	0.93 ($\pm$0.10)	0.96 ($\pm$0.05)	0.83 ($\pm$0.10)	9.4 ($\pm$1.2)	**3** ($\pm$1)	12 ($\pm$6)
	$L_{W_{\text{rw}}}$	**0.54** ($\pm$0.05)	0.91 ($\pm$0.08)	0.92 ($\pm$0.09)	0.96 ($\pm$0.06)	0.83 ($\pm$0.13)	10.7 ($\pm$1.4)	**3** ($\pm$1)	12 ($\pm$6)
Boyd *et al.*	$L_{W_{\text{sym}}}$	**0.54** ($\pm$0.05)	0.93 ($\pm$0.08)	0.97 ($\pm$0.06)	0.97 ($\pm$0.06)	0.88 ($\pm$0.13)	6.9 ($\pm$1.2)	**3** ($\pm$1)	11 ($\pm$6)
	$L_{W_{\text{rw}}}$	**0.54** ($\pm$0.05)	0.93 ($\pm$0.08)	0.97 ($\pm$0.06)	0.97 ($\pm$0.06)	0.89 ($\pm$0.13)	7.3 ($\pm$1.1)	**3** ($\pm$1)	12 ($\pm$6)
Spectral clustering		0.49 ($\pm$0.02)	0.87 ($\pm$0.01)	0.95 ($\pm$0.01)	0.95 ($\pm$0.01)	0.83 ($\pm$0.01)	11.6 ($\pm$1.5)	**3** ($\pm$0)	–

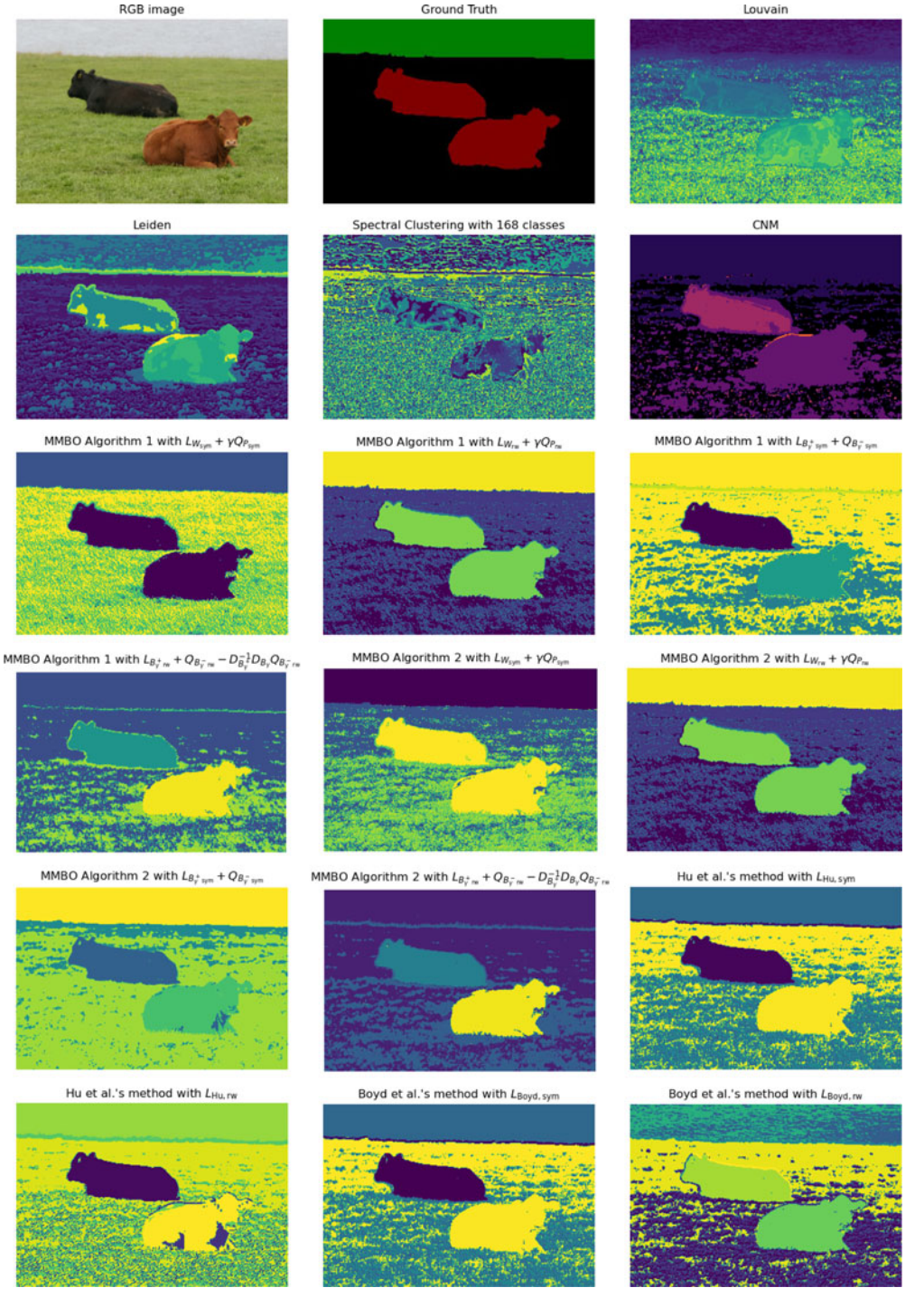


Figure 7. *The 'two cows' image segmented using different methods with* $\gamma = 1$. *The number of clusters* K *used by MMBO algorithms, Hu et al.'s method and Boyd et al.'s method is obtained from Louvain's method, that is,* $K = 168$. *Moreover, for the MMBO schemes, Hu et al.'s method and Boyd et al.'s method, we choose* $m = K = 168$. *Each method's displayed image segmentation result is the one with the highest modularity scores for that method from among* 20 *runs.*

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press



Figure 8. *The 'two cows' image is segmented using different methods with* $\gamma = 1$. *The number of clusters K used by MMBO algorithms, Hu et al.'s method and Boyd et al.'s method is obtained from the ground truth (shown in Figure 7), that is,* $K = 3$. *Moreover, for the MMBO scheme, Hu et al.'s method and Boyd et al.'s method, we choose* $m = K = 3$. *Each method's displayed image segmentation result is the one with the highest modularity scores for that method from among* 20 *runs.*

from the average in parentheses. The MMBO schemes, Hu *et al.*'s method and Boyd *et al.*'s method are implemented using the modularity-based stopping criterion (49). Moreover, we set $m = K$.

These figures also explore two different strategies for determining the value of K in spectral clustering, MMBO schemes, Hu *et al.*'s method and Boyd *et al.*'s method. The first strategy involves using the Louvain method to determine K (Figure 7). We see that spectral clustering is not capable of segmenting the cows completely. Even worse, the Louvain method segments the images too finely if reproducing the ground truth is the goal, finding about 168 (non-empty) clusters. This does, however, give a high

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

modularity score. This suggests that there are better values of γ to choose for the Louvain method if we want to reproduce the ground truth partition. Besides with $\gamma = 1$, we also tested Louvain's method with $\gamma \in \{0.5, 1.5, 2\}$ and found on average 72 (for $\gamma = 0.5$), 263 (for $\gamma = 1.5$) and 359 (for $\gamma = 2$) non-empty clusters.[61]

In Figure 7, the MMBO schemes effectively group most sky-representing pixels into a single class, while the Boyd method (with $L_{W_{\text{rw}}}$) manages to cluster some cow-representing pixels together. However, none of these methods achieves a level of segmentation that comes close to matching the ground truth, although it should be noted that the MBO-based methods return a number of non-empty clusters that is much closer to the ground truth than might be expected based on the choice $K = 168$.

The second strategy for determining the value of K is based on the ground truth. Upon examining the ground truth, we find that $K = 3$. This approach leads to the results shown in Figure 8. In this figure, the MMBO schemes (with $L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$ and $L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$) successfully classify the sky and cows, clustering the cows (of different colours) into a single category. Nevertheless, a few pixels representing grass are incorrectly labelled as cows.

In Tables 14–16, we present quantitative results. With respect to the modularity score, the Louvain method performs best, but at substantially greater run time than most other methods (except CNM). It should be noted that the Louvain method finds 168 clusters. Out of the other methods, the Leiden algorithm achieves the highest average modularity and shortest (average) running time, but ARI, inverse purity and NMI are lower than those of MBO-based methods. Also the number of non-empty clusters that is found by the Leiden algorithm, although much closer to the ground truth than the number that the Louvain method found, is still further removed from the ground truth number than the numbers obtained by the MBO-based methods. This suggests modularity may not be the best metric to capture the ground truth behaviour in this case; this suggestion is all but confirmed by the low modularity score obtained by the ground truth in Table 14, at least for the modularity score with $\gamma = 1$ and the NG null model. In Table 16, we show the result with the ground truth value $K = 3$, for those methods that allow us to specify the value of K. Observing Tables 15 and 16, we note that in some cases, the MMBO schemes (with $L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$) obtain a slightly higher average modularity score than the methods from Hu *et al.* and Boyd *et al.*

8. Conclusion and future research

In this paper, we have derived a novel expression for the modularity function at a fixed number of communities in terms of total variation functional based on the graph and a signless total variation functional based on the null model. From this expression, we have developed a modularity MBO (MMBO) approach for modularity optimisation. When working with large networks, we implement the Nyström extension with QR decomposition to compute the leading eigenvalues and corresponding eigenvectors.

Our MMBO schemes can handle large data sets (such as the MNIST data set) while requiring low computational costs. In numerical experiments, we compared our method with the Louvain method [6], the Leiden algorithm [71], CNM [19] and spectral clustering [37], and the methods by Hu *et al.* [33] and Boyd *et al.* [7]. These experiments show that our methods are competitive in terms of modularity scores and run times with most of the other methods. We have observed that the Leiden algorithm often obtains a somewhat higher modularity score in a shorter time. With respect to the other evaluation metrics of interest, the MMBO methods sometimes outperform all the other methods. In particular, we note that all the MBO-based methods, and especially the MMBO methods, often find substantially smaller numbers of clusters than the other methods, including Leiden, which hints at an inherent scale in these methods that is of interest for future research.

Other potential directions for future research are the generalisation of the MMBO algorithms to signed graphs, that is, graphs in which the edge weights may be negative as well, along the lines of

[61] In fact, the results of Hu *et al.* [34] suggest that values around $\gamma = 0.13$ could lead to a smaller number of clusters, closer to the ground truth.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Cucuringu *et al.* [20], the incorporation of mass constraints or fidelity forcing based on training data, as in Budd and Van Gennip [11] and Budd *et al.* [12], respectively, and the combination of the MMBO scheme with artificial neural networks, similar to Liu *et al.* [45]. Also the use of different null models can be considered.

The newly proposed MMBO algorithms in this paper share an underlying philosophy in their construction with the methods by Hu *et al.* and Boyd *et al.*. The first step in devising each of these methods is to rewrite the modularity functional into an equivalent form. Then the non-convex discrete domain is relaxed into a convex domain. The first step in this approach allows for the use of a great variety of functionals that are all equivalent to the modularity functional on the original discrete domain. One of the reasons for the choice we made in this paper is that it clearly illustrates the role of the null model in the modularity functional. A more systematic study into the effect of the choice of equivalent functional in this first step on the accuracy of the resulting method would be a very illuminating topic for future research.

Acknowledgements. For a significant period during the early development of this paper, ZL was affiliated to the Department of Mathematics and Computer Science, Freie Universität Berlin.

For part of the period during which this work was written, YvG acknowledges support from the European Union Horizon 2020 research and innovation programme under Marie Skłodowska-Curie grant agreement No. 777826 (NoMADS).

The authors thank both anonymous reviewers for their valuable feedback on an earlier manuscript. It has led to a significantly improved final version.

Competing interests. None.

References

[1] Ambrosio, L., Gigli, N. & Savaré, G. (2008). *Gradient Flows: In Metric Spaces and in the Space of Probability Measures*. second ed. Springer Science & Business Media.

[2] Arenas, A., Fernández, A. & Gómez, S. (2008) Analysis of the structure of complex networks at different resolution levels. *New J. Phys.* **10**(5), 053039.

[3] Aynaud, T. (2020). Python-louvain x.y: Louvain algorithm for community detection. Available at: https://github.com/taynaud/python-louvain, The package name on pip is python-louvain but it is imported as community in python,

[4] Bertozzi, A. L. & Flenner, A. (2012) Diffuse interface models on graphs for classification of high dimensional data. *Multiscale Model. Sim.* **10**(3), 1090–1118.

[5] Bertozzi, A. L. & Flenner, A. (2016) Diffuse interface models on graphs for classification of high dimensional data. *Siam Rev.* **58**(2), 293–328.

[6] Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. (2008) Fast unfolding of communities in large networks. *J. Stat. Mech.: Theor. Exp.* **2008**(10), P10008.

[7] Boyd, Z. M., Bae, E., Tai, X.-C. & Bertozzi, A. L. (2018) Simplified energy landscape for modularity using total variation. *SIAM J. Appl. Math.* **78**(5), 2439–2464.

[8] Braides, A. (2002). Γ*-Convergence for Beginners*. vol. 22 of Oxford Lecture Series in Mathematics and its Applications, first ed. Oxford University Press, Oxford.

[9] Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z. & Wagner, D. (2007) On modularity clustering. *IEEE T. Knowl. Data En.* **20**(2), 172–188.

[10] Budd, J. (2023). Graph-based learning for imageprocessing. accessed 29-August-2023. *Available at:* https://jeremybudd.com/.

[11] Budd, J. M. & Van Gennip, Y. (2022) Mass-conserving diffusion-based dynamics on graphs. *Euro. J. Appl. Math.* **33**(3), 423–471.

[12] Budd, J., van Gennip, Y. & Latz, J. (2021) Classification and image processing with a semi-discrete scheme for fidelity forced Allen–Cahn on graphs, *GAMM Mitteilungen Special Issue: Scientific Machine Learning Part-I*, Vol. **44**, pp. 1–43.

[13] Bunch, J. R., Nielsen, C. P. & Sorensen, D. C. (1978) Rank-one modification of the symmetric eigenproblem. *Numer. Math.* **31**(1), 31–48.

[14] Butcher, J. C. (2016). *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons.

[15] Chacón, Jé E. & Rastrojo, A. I. (2023) Minimum adjusted rand index for two clusterings of a given size. *Adv. Data. Anal. Classif.* **17**(1), 125–133. https://doi.org/10.1007/s11634-022-00491-w

[16] Chung, Fan R. K. (1997). Spectral Graph Theory. In: CBMS Regional Conference Series in Mathematics, *Conference Board of the Mathematical Sciences*, Washington, DC, Providence, RI: by the American Mathematical Society, vol. **92**.

[17] Clarke, F. H. (1983) *Optimization and Nonsmooth Analysis. Canadian Mathematical Society Series of Monographs and Advanced Texts, A Wiley-Interscience Publication*, John Wiley & Sons, Inc, New York.

[18] Clason, C. (2022). Nonsmooth analysis and optimization. https://arxiv.org/abs/1708.04180.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

[19] Clauset, A., Newman, M. E. J. & Moore, C. (2004) Finding community structure in very large networks. *Phys. Rev. E.* **70**(6), 066111.

[20] Cucuringu, M., Pizzoferrato, A. & van Gennip, Y. (2021) An MBO scheme for clustering and semi-supervised clustering of signed networks. *Commun. Math. Sci* **19**(1), 73–109. https://dx.doi.org/10.4310/CMS.2021.v19.n1.a4.

[21] Duch, J. & Arenas, A. (2005) Community detection in complex networks using extremal optimization. *Phys. Rev. E.* **72**(2), 027104.

[22] Fortunato, S. & Barthélemy, M. (2007) Resolution limit in community detection. *Proc. Nat. Acad. Sci.* **104**(1), 36–41. www.pnas.org/cgi/doi/10.1073/pnas.0605965104.

[23] Fowlkes, C., Belongie, S., FanC. & Malik, J. (2004) Spectral grouping using the yström method. *IEEE T. Pattern. Anal.* **26**(2), 214–225.

[24] Garcia-Cardona, C., Merkurjev, E., Bertozzi, A. L., Flenner, A. & Percus, A. G. (2014) Multiclass data segmentation using diffuse interface methods on graphs. *IEEE T. Pattern. Anal.* **36**(8), 1600–1613.

[25] Gates, A. J. & Ahn, Y.-Y. (2017) The impact of random models on clustering similarity. *J. Mach. Learn. Res.* **18**(87), 1–28.

[26] Girvan, M. & Newman, M. E. J. (2002) Community structure in social and biological networks. *Proc. Nat. Acad. Sci.* **99**(12), 7821–7826.

[27] Golub, G. H. & Van Loan, C. F. (2013). *Matrix Computations. Johns Hopkins Studies in the Mathematical Sciences*. fourth ed. Johns Hopkins University Press, Baltimore, MD.

[28] Guimerà, R., Sales-Pardo, M. & Nunes Amaral, L. A. (2004) Modularity from fluctuations in random graphs and complex networks. *Phys. Rev. E.* **70**(2), 025101.

[29] Hale, J. K. (2009). *Ordinary Differential Equations*, Courier Corporation.

[30] Hall, B. C. (2003). *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*, volume 27, Springer,

[31] Hoffman, K. & Kunze, R. (1971) *Linear Algebra*. second ed. Prentice-Hall, Inc, Englewood Cliffs, NJ.

[32] Holland, P. W., Laskey, K. B. & Leinhardt, S. (1983) Stochastic blockmodels: First steps. *Soc. Networks* **5**(2), 109–137.

[33] Hu, H., Laurent, T., Porter, M. A. & Bertozzi, A. L. (2013) A method based on total variation for network modularity optimization using the MBO scheme. *SIAM J. Appl. Math.* **73**(6), 2224–2246.

[34] Hu, H., van Gennip, Y., Hunter, B., Bertozzi, A. L. & Porter, M. A. (2012). Multislice modularity optimization in community detection and image segmentation. In *2012 IEEE 12th International Conference on Data Mining Workshops*, pp. 934–936.

[35] Hubert, L. & Arabie, P. (1985) Comparing partitions. *J. Classif.* **2**(1), 193–218.

[36] Jeub, L. G. S., Sporns, O. & Fortunato, S. (2018) Multiresolution consensus clustering in networks. *Sci. Rep.* **8**(1), 1–16.

[37] Jianbo Shi & Malik, J. (2000) Normalized cuts and image segmentation. *IEEE T. Pattern. Anal.* **22**(8), 888–905.

[38] Karataş, A. & Şahin, S. (2018) Application areas of community detection: A review. In: 2018 International Congress On Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), IEEE, pp. 65–70.

[39] Keetch, B. & van Gennip, Y. (2019) A max-cut approximation using a graph based MBO scheme. *Discrete Contin. Dyn. Syst. Series B* **24**(11), 6091–6139.

[40] Lancichinetti, A. & Fortunato, S. (2009) Community detection algorithms: A comparative analysis. *Phys. Rev. E.* **80**(5), 056117.

[41] Lancichinetti, A. & Fortunato, S. (2011) Limits of modularity maximization in community detection. *Phys. Rev. E.* **84**(6), 066122.

[42] Lancichinetti, A., Fortunato, S. & Radicchi, F. (2008) Benchmark graphs for testing community detection algorithms. *Phys. Rev. E.* **78**(4), 046110.

[43] LeCun, Y., Cortes, C. & Christopher, J. C. B. 1998). The MNIST database of handwritten digits.

[44] Lehoucq, R. B., Sorensen, D. C. & Yang, C. (1998) ARPACK users' guide: Solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods. *SIAM*.

[45] Liu, H., Liu, J., Chan, R. & Tai, X.-C. (2023) Double-well net for image segmentation. https://arxiv.org/abs/2401.00456.

[46] Luo, X. & Bertozzi, A. L. (2017) Convergence of the graph Allen–Cahn scheme. *J. Stat. Phys.* **167**, 934–958. https://doi.org/10.1007/s10955-017-1772-4.

[47] Maso, G. D. (1993). An introduction to Γ-convergence, *Progress in Nonlinear Differential Equations and Their Applications*, Vol. **8**, first ed. Birkhäuser, Boston

[48] Merkurjev, E., Garcia-Cardona, C. & Bertozzi, A. L., Flenner, A. & Percus, A. G. (2014) Diffuse interface methods for multiclass segmentation of high-dimensional data. *Appl. Math. Lett.* **33**, 29–34. https://doi.org/10.1016/j.aml.2014.02.008.

[49] Merkurjev, E., Kostić, T. & Bertozzi, A. L. (2013) An MBO scheme on graphs for classification and image processing. *SIAM J. Imaging. Sci.* **6**(4), 1903–1930.

[50] Merriman, B., Bence, J. K. & Osher, S. J. (1992) Diffusion generated motion by mean curvature. UCLA Department of Mathematics CAM report 92-18.

[51] Merriman, Barry, Bence, James K. & Osher, Stanley J. (1993) Diffusion generated motion by mean curvature, *AMS Selected Letters*, Crystal Grower's Workshop, pp. 73–83.

[52] Microsoft. Microsoft research cambridge object recognition image database version 1.0., https://www.microsoft.com/en-us/download/details.aspx?id=52644, 18 May 2005.

[53] Mollaian, M., Dörgő, G. & Palazoglu, A. (2021) Studying the synergy between dimension reduction and clustering methods to facilitate fault classification, *Computer Aided Chemical Engineering*, Vol. **50**, Elsevier, pp. 819–824.

[54] Mucha, P. J., Richardson, T., Macon, K., Porter, M. A. & Onnela, J.-P. (2010) Community structure in time-dependent, multiscale, and multiplex networks. *Science* **328**, 876–878. https://doi.org/10.1126/science.1184819

[55] Newman, M. (2010). *Networks: An Introduction*, 1st edition, Oxford University Press. https://doi.org/10.1093/acprof:oso/9780199206650.001.0001

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

[56] Newman, M. E. J. (2006) Modularity and community structure in networks. *Proc. Natl. Acad. Sci.* **103**(23), 8577–8582. www.pnas.org/cgi/doi/10.1073/pnas.0601602103.
[57] Newman, M. E. J. & Girvan, M. (2004) Finding and evaluating community structure in networks American Physical Society. *Phys. Rev. E.* **69**(2), 026113. https://link.aps.org/doi/10.1103/PhysRevE.69.026113.
[58] Nyström, E. J. (1930) Über die praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. *Acta. Math.* **54**(0), 185–204.
[59] Pedregosa, Fabian, Varoquaux, Gaël, Gramfort, Alexandre, et al. (2011) Scikit-learn: Machine learning in Python. *J Mach Learn Res* **12**, 2825–2830
[60] Perlasca, P., Frasca, M., Ba, C. T., Gliozzo, J., Notaro, M., Pennacchioni, M., Valentini, G., Mesiti, M. & Cherifi, H. (2020) Multi-resolution visualization and analysis of biomolecular networks through hierarchical community detection and web-based graphical tools. *PLoS ONE* **15**(12), e0244241.
[61] Rand, W. M. (1971) Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **66**(336), 846–850.
[62] Reichardt, Jörg, Bornholdt, S. (2006) Statistical mechanics of community detection. *Phys. Rev. E.* **74**(1), 016110.
[63] Schütze, H., Manning, C. D. & Raghavan, P. (2008). *Introduction to Information Retrieval*, volume 39, Cambridge University Press Cambridge, .
[64] Shannon, C. E. (1948) A mathematical theory of communication. *Bell Sys. Tech. J.* **27**(3), 379–423.
[65] Singh, D. K. & Choudhury, P. (2023) Community detection in large-scale real-world networks, *Advances in Computers*, Vol. **128**, Elsevier, pp. 329–352.
[66] Steinley, D. (2004) Properties of the Hubert-Arable Adjusted Rand Index.. *Psychol. Methods* **9**(3), 386–396.
[67] Sun, B. & Chang, H. (2022) Proximal gradient methods for general smooth graph total variation model in unsupervised learning. *J. Sci. Comput.* **93**(1), 23, Paper No. 2.
[68] The NetworkX Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks., Accessed: 2022-01-27. *Available at:* https://networkx.org/.
[69] Thompson, R. C. (1976) The behavior of eigenvalues and singular values under perturbations of restricted rank. *Linear Algebra Appl.* **13**(1), 69–78.
[70] Traag, V. (2020). The Leiden algorithm python package. *Available at:* https://github.com/vtraag/leidenalg.
[71] Traag, V. A., Waltman, L. & van Eck, N. J. (2019) From louvain to Leiden: Guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 5233.
[72] van Gennip, Y. & Bertozzi, A. L. (2012) Γ-convergence of graph Ginzburg–Landau functionals. *Adv. Differential Equ.* **11**(12), 1115–1180.
[73] van Gennip, Y., Guillen, N., Osting, B. & Bertozzi, A. L. (2014) Mean curvature, threshold dynamics, and phase field theory on finite graphs. *Milan. J. Math.* **82**(1), 3–65.
[74] von Luxburg, U. (2007) A tutorial on spectral clustering. *Stat. Comput.* **17**(4), 395–416.
[75] Weyl, H. (1912) Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung). *Math. Ann.* **71**(4), 441–479.

Appendix A Multiple-well potential with 1-norms and 2-norms

From (36) we recall that, for $w \in \mathbb{R}^K$,

$$\Phi_{\text{mul}}(w) := \frac{1}{2}\left(\prod_{k=1}^{K} \frac{1}{4}||w - e^{(k)}||_1^2\right).$$

Merkurjev *et al.* [48, Section 2.1] and Garcia-Cardona *et al.* [24, Section 3.1] argue that the choice for 1-norms instead of 2-norms is driven by the presence of an unwanted local minimiser in the interior of the simplex $\mathfrak{S}(K)$ from (44) when 2-norms are used.

First we prove that such a minimiser indeed does not exist with the current definition as in (36). Recall that $\mathbf{1}_K \in \mathbb{R}^K$ is the column vector whose entries are all 1.

Lemma A.1. *Let $K \geq 3$. The only local minimisers of the function Φ_{mul} from (36) on $\mathbb{R}^K$ are its global minimisers at the vertices of the simplex $\mathfrak{S}(K)$ from (44) and a local minimiser (which is not a global minimiser) at $-\mathbf{1}_K$.*

Proof. We recall that the vertices of the simplex $\mathfrak{S}(K)$ are the vectors $e^{(k)}$ that are defined in Section 4.2. Since $\Phi_{\text{mul}} \geq 0$ and $\Phi_{\text{mul}}(w) = 0$ if and only if $w = e^{(k)}$, it is clear that Φ_{mul} has global minima at and only at these vertices.

Now assume that $w \in \mathbb{R}^K$ is such that, for all $j \in \{1, \ldots, K\}$, $w_j \notin \{-1, 1\}$. Then

$$\frac{\partial}{\partial w_j}\frac{1}{4}\|w - e^{(k)}\|_1^2 = \frac{1}{2}\|w - e^{(k)}\|_1 \frac{\partial}{\partial w_j}\sum_{l=1}^{K}|w_l - e_l^{(k)}| = \frac{1}{2}\text{sgn}(w_j - e_j^{(k)})\|w - e^{(k)}\|_1.$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Thus

$$\frac{\partial}{\partial w_j}\Phi_{\text{mul}}(w) = \frac{1}{4}\sum_{k=1}^{K}\text{sgn}(w_j - e_j^{(k)})\|w - e^{(k)}\|_1 \prod_{\substack{l=1 \\ l\neq k}}^{K}\frac{1}{4}\|w - e^{(l)}\|_1^2$$

$$= \frac{1}{4}\sum_{k=1}^{K}\text{sgn}(w_j - e_j^{(k)})\left(\frac{2\Phi_{\text{mul}}(w)}{\frac{1}{4}\|w - e^{(k)}\|_1}\right)$$

$$= 2\Phi_{\text{mul}}(w)\sum_{k=1}^{K}\frac{\text{sgn}(w_j - e_j^{(k)})}{\|w - e^{(k)}\|_1}.$$

Assume that Φ_{mul} has a local minimum at $w = w^*$ which satisfies, for all $j \in \{1, \ldots, K\}$, $w_j^* \notin \{-1, 1\}$. Then, for all $j \in \{1, \ldots, K\}$, $\frac{\partial}{\partial w_j^*}\Phi_{\text{mul}}(w) = 0$. Since w^* is not a vertex of the simplex, $\Phi_{\text{mul}}(w^*) \neq 0$ and thus, for all $j \in \{1, \ldots, K\}$,

$$\sum_{k=1}^{K}\frac{\text{sgn}(w_j^* - e_j^{(k)})}{\|w^* - e^{(k)}\|_1} = 0. \tag{60}$$

If $w_j^* < -1$, then, for all $k \in \{1, \ldots, K\}$, $\text{sgn}(w_j^* - e_j^{(k)}) = -1$ and thus

$$\sum_{k=1}^{K}\frac{-1}{\|w^* - e^{(k)}\|_1} = 0.$$

This is a contradiction. If $w_j^* > 1$, then, for all $k \in \{1, \ldots, K\}$, $\text{sgn}(w_j^* - e_j^{(k)}) = 1$, which leads to a similar contradiction. Hence, it must hold that, for all $j \in \{1, \ldots, K\}$, $w_j^* \in (-1, 1)$. In that case, for all $j \in \{1, \ldots, K\}$, if $k = j$ we have $\text{sgn}(w_j^* - e_j^{(k)}) = -1$ and for all $k \in \{1, \ldots, K\} \setminus \{j\}$ we have $\text{sgn}(w_j^* - e_j^{(k)}) = 1$. Therefore, for all $j \in \{1, \ldots, K\}$,

$$\frac{1}{\|w^* - e^{(j)}\|_1} = \sum_{\substack{k=1 \\ k\neq j}}^{K}\frac{1}{\|w^* - e^{(k)}\|_1}.$$

Summing both sides over all $j \in \{1, \ldots, K\}$, we obtain

$$\sum_{j=1}^{K}\frac{1}{\|w^* - e^{(j)}\|_1} = \sum_{j=1}^{K}\sum_{\substack{k=1 \\ k\neq j}}^{K}\frac{1}{\|w^* - e^{(k)}\|_1} = (K-1)\sum_{k=1}^{K}\frac{1}{\|w^* - e^{(k)}\|_1}.$$

Since $K \geq 3$, it follows that

$$\sum_{k^*=1}^{K}\frac{1}{\|w^* - e^{(k)}\|_1} = 0,$$

which is again a contradiction. (For future reference, we note that the proof by contradiction that followed (60) did not depend on any properties of $\|w^* - e^{(k)}\|_1$ except its positivity.)

Hence if Φ_{mul} has a local minimum at w^*, there must exist a $j \in \{1, \ldots, K\}$, such that $w_j^* \in \{-1, 1\}$. Assume there is exactly one such j which we call j^* and define the subspace

$$\mathbb{R}_{j^*}^K := \{w \in \mathbb{R}^K : w_{j^*} = w_{j^*}^*\}.$$

Since Φ_{mul} has a local minimum at w^*, $\Phi_{\text{mul}}|_{\mathbb{R}_{j^*}^K}$ also has a local minimum at w^*. If we write

$$\|w - e^{(k)}\|_{1,\mathbb{R}_{j^*}^K} := \sum_{\substack{j=1 \\ j\neq j^*}}^{K}|w_j - e_j^{(k)}|,$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

then

$$\|w - e^{(k)}\|_1\big|_{\mathbb{R}^K_{j^*}} = q_k + \|w - e^{(k)}\|_{1,\mathbb{R}^K_{j^*}},$$

where

$$q_k := (1 + w^*_{j^*}) - 2w^*_{j^*}\delta_{kj^*} = \begin{cases} 0, & = \text{if } (w^*_{j^*} = -1 \text{ and } k \neq j^*) \text{ or } (w^*_{j^*} = 1 \text{ and } k = j^*), \\ 2, & = \text{if } (w^*_{j^*} = -1 \text{ and } k = j^*) \text{ or } (w^*_{j^*} = 1 \text{ and } k \neq j^*). \end{cases}$$

In particular $q_k \geq 0$. Thus

$$\Phi_{\text{mul}}|_{\mathbb{R}^K_{j^*}}(w) = \frac{1}{2}\left(\prod_{k=1}^{K}\frac{1}{4}\left(||w - e^{(k)}||_1\big|_{\mathbb{R}^K_{j^*}}\right)^2\right) = \frac{1}{2}\left(\prod_{k=1}^{K}\frac{1}{4}\left(q_k + ||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}}\right)^2\right).$$

Recall that we have assumed that, if $j \neq j^*$, then $w^*_j \notin \{-1, 1\}$. Hence, in that case

$$\begin{aligned}\frac{\partial}{\partial w_j}\frac{1}{4}\left(q_k + ||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}}\right)^2 &= \frac{1}{2}\left(q_k + ||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}}\right)\frac{\partial}{\partial w_j}||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}} \\ &= \frac{1}{2}\left(q_k + ||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}}\right)\text{sgn}(w_j - e^{(k)}_j)\end{aligned}$$

and therefore

$$\begin{aligned}\frac{\partial}{\partial w_j}\Phi_{\text{mul}}|_{\mathbb{R}^K_{j^*}}(w) &= \frac{1}{2}\sum_{k=1}^{K}\frac{1}{2}\text{sgn}(w_j - e^{(k)}_j)\left(q_k + ||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}}\right)\prod_{\substack{l=1\\l\neq k}}^{K}\frac{1}{4}\left(q_l + ||w - e^{(l)}||_{1,\mathbb{R}^K_{j^*}}\right)^2 \\ &= \frac{1}{2}\sum_{k=1}^{K}\text{sgn}(w_j - e^{(k)}_j)\frac{\Phi_{\text{mul}}|_{\mathbb{R}^K_{j^*}}(w)}{q_k + ||w - e^{(k)}||_{1,\mathbb{R}^K_{j^*}}}.\end{aligned}$$

Since $\Phi_{\text{mul}}|_{\mathbb{R}^K_{j^*}}$ has a local minimum at w^*, we require that, for all $j \in \{1, \ldots, K\} \setminus \{j^*\}$, $\frac{\partial}{\partial w_j}\Phi_{\text{mul}}|_{\mathbb{R}^K_{j^*}}(w^*) = 0$. Since $q_k + ||w^* - e^{(k)}||_{1,\mathbb{R}^K_{j^*}} > 0$, we can use a similar proof by contradiction as that which followed (60) to show that there must be a $j^{**} \in \{1, \ldots, K\} \setminus \{j^*\}$ for which $w^*_{j^{**}} \in \{-1, 1\}$. If, for all $j \in \{1, \ldots, K\} \setminus \{j^*, j^{**}\}$, $w^*_j \notin \{-1, 1\}$, then via essentially the same arguments as in the previous case (with two values $q_k, q_{k'} \in \{0, 2\}$, $k \neq k'$; in particular their sum is non-negative), it can be shown that w^* is also not a local minimiser, and thus there must be another component of w^* in $\{-1, 1\}$. Repeating this approach further shows that w^* is not a local minimiser if at least one of its components is not in $\{-1, 1\}$.

Consider now the case that all components of w^* are in $\{-1, 1\}$ and not all of its components are -1. Thus, at least two components are 1 since it is assumed that w^* is not a vertex. We define

$$J := \{j \in \{1, \ldots, K\} : w^*_j = 1\};$$

then

$$\|w^* - e^{(k)}\|_1 = |w^*_k - 1| + 2|J \setminus \{k\}|.$$

Since w^* is not one of the vertices $e^{(k)}$, we have, for all $k \in \{1, \ldots, K\}$, $|w^*_k - 1| = 2$ or $|J \setminus \{k\}| \geq 1$, thus $\|w^* - e^{(k)}\|_1 \geq 2$.

If there exists a k^* such that $w^*_{k^*} = 1$, and thus $|J \setminus \{k^*\}| \geq 1$, let $l^* \in J \setminus \{k^*\}$ and, for all $\varepsilon \in (0, 2)$, define w^ε via $w^\varepsilon_j := w^*_j$ if $j \neq l^*$ and $w^\varepsilon_j := w^*_j - \varepsilon$ if $j = l^*$. Then

$$\|w^\varepsilon - e^{(k^*)}\|_1 = 2|J \setminus \{k^*, l^*\}| + (2 - \varepsilon) = 2|J \setminus \{k^*\}| - \varepsilon = \|w^* - e^{(k^*)}\|_1 - \varepsilon,$$

$$\|w^\varepsilon - e^{(l^*)}\|_1 = |w^\varepsilon_{l^*} - 1| + 2|J \setminus \{l^*\}| = \varepsilon + 2|J \setminus \{l^*\}| = \|w^* - e^{(l^*)}\|_1 + \varepsilon$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

and, for all $k \in \{l, \dots, K\} \setminus \{k^*, l^*\}$,

$$\|w^\varepsilon - e^{(k)}\|_1 = |w_k^\varepsilon - 1| + 2|J \setminus \{k, l^*\}| + |w_{l^*}^\varepsilon + 1| = |w_k^\varepsilon - 1| + 2|J \setminus \{k, l^*\}| + 2 - \varepsilon$$
$$= |w_k^\varepsilon - 1| + 2|J \setminus \{k, l^*\}| + |w_{l^*}^* + 1| - \varepsilon = \|w^* - e^{(k)}\|_1 - \varepsilon.$$

For notational convenience, we write $a_k := \|w^* - e^{(k)}\|_1$. Then

$$\Phi_{\text{mul}}(w^\varepsilon) = 2^{-2K-1} f^2(\varepsilon), \quad \text{with} \quad f(\varepsilon) := (a_{l^*} + \varepsilon) \prod_{\substack{k=1 \\ k \neq l^*}}^{K} (a_k - \varepsilon).$$

We compute

$$f'(\varepsilon) = \prod_{\substack{k=1 \\ k \neq l^*}}^{K} (a_k - \varepsilon) + (a_{l^*} + \varepsilon) \sum_{\substack{k=1 \\ k \neq l^*}}^{K} (-1) \prod_{\substack{l=1 \\ l \neq l^* \\ l \neq k}}^{K} (a_l - \varepsilon) = \prod_{\substack{l=1 \\ l \neq l^*}}^{K} (a_l - \varepsilon) \left(1 - (a_{l^*} + \varepsilon) \sum_{\substack{k=1 \\ k \neq l^*}}^{K} \frac{1}{a_k - \varepsilon} \right).$$

Since $a_{l^*} = a_{k^*}$, we have $\frac{a_{l^*} + \varepsilon}{a_{k^*} - \varepsilon} > 1$ and thus $f'(\varepsilon) < 0$ for $\varepsilon \in (0, 2)$. Thus, for small ε, $\Phi_{\text{mul}}(w^\varepsilon) < \Phi_{\text{mul}}(w^*)$ which contradicts w^* being a local minimiser of Φ_{mul}.

It remains to study the case in which, for all $k \in \{1, \dots, K\}$, $|w_k^* - 1| = 2$, that is, $w^* = -\mathbf{1}_K$. We claim this is a local, but not global minimiser of Φ_{mul}. That $-\mathbf{1}_K$ is not a global minimiser follows simply from $\Phi_{\text{mul}}(-\mathbf{1}_K) = \frac{1}{2} > 0$.

To prove that $-\mathbf{1}_K$ is a local minimiser via a proof by contradiction, assume that for all $\varepsilon \in (0, 2)$, there exists a $\tilde{z}^\varepsilon \in \mathbb{R}^K$ such that $\|\tilde{z}^\varepsilon + \mathbf{1}_K\|_1 \leq \varepsilon$ and $\Phi_{\text{mul}}(\tilde{z}^\varepsilon) < \Phi_{\text{mul}}(-\mathbf{1}_K)$. Thus for every $\varepsilon \in (0, 2)$ the set

$$\operatorname*{argmin}_{\substack{z \in \mathbb{R}^K \\ \|z + \mathbf{1}_K\|_1 \leq \varepsilon}} \Phi_{\text{mul}}(z) \tag{61}$$

is not empty and does not contain $-\mathbf{1}_K$. For every $\varepsilon \in (0, 2)$, let z^ε be an element of this set. Then there must be a k^* such that $\|z^\varepsilon - e^{(k^*)}\|_1 < \|-\mathbf{1}_K - e^{(k^*)}\|_1 < 2$ and therefore there exists an $\eta \in (0, 2]$ such that $\|z^\varepsilon - e^{(k^*)}\|_1 = 2 - \eta$. Hence

$$|z_{k^*}^\varepsilon - 1| + \sum_{\substack{j=1 \\ j \neq k^*}}^{K} |z_j^\varepsilon + 1| = 2 - \eta. \tag{62}$$

For future use, we note that

$$2 = \|-\mathbf{1}_K - e^{(k^*)}\|_1 \leq \|-\mathbf{1}_K - z^\varepsilon\|_1 + \|z^\varepsilon - e^{(k^*)}\|_1 \leq \varepsilon + 2 - \eta$$

and thus $0 \leq \eta \leq \varepsilon$.

Because, for all $j \in \{1, \dots, K\}$, $z_j^\varepsilon \in [-1 - \varepsilon, -1 + \varepsilon]$, there exist $\varepsilon_j \in [-\varepsilon, \varepsilon]$, such that, for all $j \in \{1, \dots, K\}$, $z_j^\varepsilon = -1 + \varepsilon_j$. Condition (62) implies

$$2 - \varepsilon_{k^*} + \sum_{\substack{j=1 \\ j \neq k^*}}^{K} |\varepsilon_j| = 2 - \eta,$$

thus

$$\varepsilon_{k^*} = \eta + \sum_{\substack{j=1 \\ j \neq k^*}}^{K} |\varepsilon_j|.$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

In particular, $\varepsilon_{k^*} > 0$. Furthermore, for $l \neq k^*$,

$$\|z^\varepsilon - e^{(l)}\|_1 = |z_l - 1| + \sum_{\substack{j=1 \\ j\neq l}}^{K} |z_j + 1| = 2 - \varepsilon_l + \sum_{\substack{j=1 \\ j\neq l}}^{K} |\varepsilon_j| = 2 - \varepsilon_l + \varepsilon_{k^*} + \sum_{\substack{j=1 \\ j\neq k^*}}^{K} |\varepsilon_j| - |\varepsilon_l|$$

$$= 2 + \eta - \varepsilon_l - |\varepsilon_l| + 2\sum_{\substack{j=1 \\ j\neq k^*}}^{K} |\varepsilon_j| = 2 + \eta - \varepsilon_l + |\varepsilon_l| + 2\sum_{\substack{j=1 \\ j\neq k^* \\ j\neq l}}^{K} |\varepsilon_j| = 2 + \eta + 2\sum_{\substack{j=1 \\ j\neq k^* \\ j\neq l}}^{K} |\varepsilon_j|.$$

For the final equality we used that $|\varepsilon_l| = \varepsilon_l$, which follows from the equalities in the first line in this calculation. Indeed, if $\varepsilon_l < 0$ then $\|z^\varepsilon - e^{(l)}\|_1$ is larger than if $\varepsilon_l \geq 0$ and this choice does not influence $\|z^\varepsilon - e^{(k)}\|_1$ for $k \neq l$ as those norms depend on ε_l only through their dependence on $|\varepsilon_l|$. Thus $\Phi_{\text{mul}}(z^\varepsilon)$ will not be minimal in the sense of (61) if $\varepsilon_l < 0$. Hence, for all $l \in \{1, \ldots, K\} \setminus \{k^*\}$, $\varepsilon_l \geq 0$. It follows that, for all $l \in \{1, \ldots, K\} \setminus \{k^*\}$,

$$\|z^\varepsilon - e^{(l)}\|_1 \geq 2 + \eta$$

and therefore

$$\Phi_{\text{mul}}(z^\varepsilon) = \frac{1}{2} g^2(\varepsilon_1, \ldots, \varepsilon_K)$$

with

$$g(\varepsilon_1, \ldots, \varepsilon_K) := 2^{-K} \|z^\varepsilon - e^{(k^*)}\|_1 \prod_{\substack{l=1 \\ l\neq k^*}}^{K} \|z^\varepsilon - e^{(l)}\|_1$$

$$\geq 2^{-K}(2 - \eta)(2 + \eta)^{K-1} = 2^{-K}(2 - \eta)(2 + \eta)^2 2^{K-3}$$

$$= 2^{-K}\left[8 + \eta(-\eta^2 - 2\eta + 4)\right] 2^{K-3},$$

where we used that $K \geq 3$. If $\eta \in \left(0, \sqrt{5} - 1\right)$, then $8 + \eta(-\eta^2 - 2\eta + 4) > 8$ and thus $g(\varepsilon_1, \ldots, \varepsilon_K) > 1$ and $\Phi_{\text{mul}}(z^\varepsilon) > \frac{1}{2} = \Phi_{\text{mul}}(-\mathbf{1}_K)$. By choosing $\varepsilon < \sqrt{5} - 1$, we force $\eta < \sqrt{5} - 1$ and thus we have a contradiction with the minimality of z^ε in the sense of (61). Hence $-\mathbf{1}_K$ is a local minimiser of Φ_{mul}. □

Next we consider the following alternative multiple-well potential:

$$\tilde{\Phi}_{\text{mul}}(w) := \frac{1}{2}\left(\prod_{k=1}^{K} \frac{1}{4} ||w - e^{(k)}||_2^2\right).$$

We show that $\tilde{\Phi}_{\text{mul}}$ has a local minimiser in the interior of the simplex.

Lemma A.2. *Let $K \geq 2$. The function $\tilde{\Phi}_{mul}$ has global minimisers at the vertices of the simplex $\mathfrak{S}(K)$ from (44). It also has a local minimiser at the unique point w^* in the interior of $\mathfrak{S}(K)$ that is equidistant (in the Euclidean distance) from all vertices of $\mathfrak{S}(K)$.*

Proof. The first statement is true since $\tilde{\Phi}_{\text{mul}} \geq 0$ and, for all $k \in \{1, \ldots, K\}$, $\tilde{\Phi}_{\text{mul}}(e^{(k)}) = 0$.

Furthermore, for all $w \in \mathbb{R}^K$ and all $j \in \{1, \ldots, K\}$,

$$\frac{\partial}{\partial w_j} \frac{1}{4} \|w - e^{(k)}\|_2^2 = \frac{1}{2}(w_j - e_j^{(k)}).$$

Thus

$$\frac{\partial}{\partial w_j} \tilde{\Phi}_{\text{mul}}(w) = \frac{1}{4} \sum_{k=1}^{K} (w_j - e_j^{(k)}) \prod_{\substack{l=1 \\ l\neq k}}^{K} \frac{1}{4} \|w - e^{(l)}\|_2^2.$$

In particular, $\tilde{\Phi}_{\text{mul}}$ is differentiable on $\mathbb{R}^K$.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Let $w^* \in \mathbb{R}^K$ be the vector in the interior of $\mathfrak{S}(K)$ equidistant to all vertices of $\mathfrak{S}(K)$, that is, for all $l \in \{1, \dots, K\}$, $\sum_{k=1}^K (w_l^* - e_l^{(k)})$, thus $w_l^* = \frac{2-K}{K}$. We compute, for all $k \in \{1, \dots, K\}$,

$$\|w - e^{(k)}\|_2^2 = \left(\frac{2-K}{K} - 1\right)^2 + (K-1)\left(\frac{2-K}{K} + 1\right)^2 = 4\left(1 - \frac{1}{K}\right) =: d^2 > 0.$$

Therefore

$$\frac{\partial}{\partial w_j}\tilde{\Phi}_{\text{mul}}(w^*) = \frac{1}{4}\sum_{k=1}^K \left(\frac{2-K}{K} - e_j^{(k)}\right) \prod_{\substack{l=1 \\ l\neq k}}^K \frac{1}{4}d^2 = 4^{-K} d^{2(K-1)} \sum_{k=1}^K \left(\frac{2-K}{K} - e_j^{(k)}\right) = 0,$$

since $\sum_{k=1}^K e_j^{(k)} = 1 - (K-1) = 2 - K$.

For second partial derivatives, with $j, l \in \{1, \dots, K\}$,

$$\begin{aligned}
\frac{\partial}{\partial w_l}\frac{\partial}{\partial w_j}\tilde{\Phi}_{\text{mul}}(w) = \frac{1}{4}\sum_{k=1}^K \Bigg(& \delta_{jl} \prod_{\substack{m=1 \\ m\neq k}}^K \frac{1}{4}\|w - e^{(m)}\|_2^2 \\
& + (w_j - e_j^{(k)}) \sum_{\substack{m=1 \\ m\neq k}}^K \frac{1}{4}\left(\frac{\partial}{\partial w_l}\|w - e^{(m)}\|_2^2\right) \prod_{\substack{r=1 \\ r\notin\{k,m\}}}^K \frac{1}{4}\|w - e^{(m)}\|_2^2 \Bigg) \\
= \frac{1}{4}\sum_{k=1}^K \Bigg(& \delta_{jl} \prod_{\substack{m=1 \\ m\neq k}}^K \frac{1}{4}\|w - e^{(m)}\|_2^2 \\
& + \frac{1}{2}(w_j - e_j^{(k)}) \sum_{\substack{m=1 \\ m\neq k}}^K (w_l - e_l^{(m)}) \prod_{\substack{r=1 \\ r\notin\{k,m\}}}^K \frac{1}{4}\|w - e^{(m)}\|_2^2 \Bigg),
\end{aligned}$$

where δ_{jl} is the Kronecker delta. In the case where $K = 2$ we should interpret $\prod_{\substack{r=1 \\ r\notin\{k,m\}}}^K \frac{1}{4}\|w - e^{(m)}\|_2^2$ as 1.

Thus at $w = w^*$ we find

$$\begin{aligned}
\frac{\partial}{\partial w_l}\frac{\partial}{\partial w_j}\tilde{\Phi}_{\text{mul}}(w)\Big|_{w=w^*} &= \frac{1}{4}\sum_{k=1}^K \left(\delta_{jl} \prod_{\substack{m=1 \\ m\neq k}}^K \frac{1}{4}d^2 + \frac{1}{2}\left(\frac{2-K}{K} - e_j^{(k)}\right) \sum_{\substack{m=1 \\ m\neq k}}^K \left(\frac{2-K}{K} - e_l^{(m)}\right) \prod_{\substack{r=1 \\ r\notin\{k,m\}}}^K \frac{1}{4}d^2\right) \\
&= \frac{\alpha}{8}\sum_{k=1}^K \left(\frac{1}{2}d^2\delta_{jl} + \left(\frac{2-K}{K} - e_j^{(k)}\right) \sum_{\substack{m=1 \\ m\neq k}}^K \left(\frac{2-K}{K} - e_l^{(m)}\right)\right),
\end{aligned}$$

where $\alpha := \left(\frac{1}{4}d^2\right)^{K-2} = \left(1 - K^{-1}\right)^{K-2} > 0$.

Since $e^{(k)}$ is a vertex of $\mathfrak{S}(K)$, we know that $\sum_{k=1}^K e_j^{(k)} = -K + 2$. Moreover, direct computation tells us that, for all $j, l \in \{1, \dots, K\}$,

$$\sum_{\substack{m=1 \\ m\neq k}}^K e_l^{(m)} = -K + q_{kl} := -K + \begin{cases} 1, & \text{if } l = k, \\ 3, & \text{if } l \neq k. \end{cases}$$

Furthermore,

$$-\sum_{k=1}^K \sum_{\substack{m=1 \\ m\neq k}}^K e_l^{(m)} = \sum_{k=1}^K (K - q_{lk}) = K^2 - \sum_{k=1}^K q_{lk} = K^2 - (1 + 3(K-1)) = K^2 - 3K + 2$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

and

$$\sum_{k=1}^{K} e_j^{(k)} \sum_{\substack{m=1 \\ m\neq k}}^{K} e_l^{(m)} = -K \sum_{k=1}^{K} e_j^{(k)} + \sum_{k=1}^{K} q_{kl} e_j^{(k)} = K(K-2) + (-3K+8-4\delta_{jl}) = K^2 - 5K + 8 - 4\delta_{jl},$$

where we used that

$$q_{kl} e_j^{(k)} = \begin{cases} -1, & \text{if } l = k \text{ and } j \neq k, \\ 1, & \text{if } l = k \text{ and } j = k, \\ -3, & \text{if } l \neq k \text{ and } j \neq k, \\ 3, & \text{if } l \neq k \text{ and } j = k, \end{cases} \quad \text{and therefore} \quad \sum_{k=1}^{K} q_{kl} e_j^{(k)} = -3K + 8 - 4\delta_{jl}.$$

Combining these results with our computation of the second partial derivatives at w^* we find

$$\begin{aligned} \frac{\partial}{\partial w_l} \frac{\partial}{\partial w_j} \tilde{\Phi}_{\text{mul}}(w^*) \Big|_{w=w^*} &= \frac{\alpha}{8} \sum_{k=1}^{K} \left(\frac{1}{2} d^2 \delta_{jl} + \frac{(K-1)(2-K)^2}{K^2} - \frac{(K-1)(2-K)}{K} e_j^{(k)} \right. \\ &\qquad \left. - \frac{2-K}{K} \sum_{\substack{m=1 \\ m\neq k}}^{K} e_l^{(m)} + e_j^{(k)} \sum_{\substack{m=1 \\ m\neq k}}^{K} e_l^{(m)} \right) \\ &= \frac{\alpha}{8} \left(\frac{1}{2} d^2 K \delta_{jl} + \frac{(K-1)(2-K)^2}{K} - \frac{(K-1)(2-K)^2}{K} \right. \\ &\qquad \left. + \frac{2-K}{K} (K^2 - 3K + 2) + K^2 - 5K + 8 - 4\delta_{jl} \right) \\ &= \frac{\alpha}{2} \left(\frac{1}{8} d^2 K - 1 \right) \delta_{jl} + \frac{\alpha}{2K} = \frac{\alpha}{2} \left(\frac{1}{2} (K-1) \delta_{jl} + K^{-1} \right). \end{aligned}$$

Thus the Hessian matrix $\mathcal{H}(w^*)$ at w^* has entries

$$(\mathcal{H}(w^*))_{jl} = \frac{\alpha}{2} \left(\frac{1}{2} (K-1) \delta_{jl} + K^{-1} \right) = \frac{\alpha}{2} \left(\frac{1}{2} (K-1) + K^{-1} \right) \mathfrak{H}_{jl},$$

where the matrix $\mathfrak{H}$ has entries $\mathfrak{H}_{jl} = 1$ if $j = l$ and $\mathfrak{H}_{jl} = \beta := \frac{\frac{\alpha}{2} K^{-1}}{\frac{\alpha}{2} \left(\frac{1}{2} (K-1) + K^{-1} \right)} = \frac{2}{K(K-1)+2} \in (0,1)$ if $j \neq l$.

The eigenvalues of $\mathfrak{H}$ have the same signs as the eigenvalues of the Hessian matrix $\mathcal{H}(w^*)$, because $\frac{\alpha}{2} \left(\frac{1}{2} (K-1) + K^{-1} \right) > 0$. We will show that all these eigenvalues are positive and therefore $\tilde{\Phi}_{\text{mul}}$ has a local minimum at w^*.

We note that $\mathfrak{H} = (1-\beta) I + \beta \mathbf{1}_K \mathbf{1}_K^T$, where $I \in \mathbb{R}^{K \times K}$ is the identity matrix and $\mathbf{1}_K \in \mathbb{R}^K$ the vector with 1 as each entry. Thus, if $v \in \mathbb{R}^K$ is an eigenvector of $\mathfrak{H}$ with eigenvalue λ, then

$$(1-\beta) v + \beta \langle \mathbf{1}_K, v \rangle \mathbf{1}_K = \lambda v,$$

which is equivalent to

$$(\lambda + \beta - 1) v = \beta \langle \mathbf{1}_K, v \rangle \mathbf{1}_K. \tag{63}$$

Direct verification shows that $v_1 := \mathbf{1}_K$ is an eigenvector with eigenvalue $\lambda_1 := \beta(K-1) + 1$. If $\{v_2, \ldots, v_K\}$ is an orthogonal basis for $(\text{span}(\{v_1\}))^\perp$, then, for all $l \in \{2, \ldots, K\}$, v_l is an eigenvector with eigenvalue $\lambda_l := 1 - \beta$, because $\langle \mathbf{1}_K, v \rangle = 0$ forces the term in parentheses in equation (63) to be zero. Since $0 < \beta < 1$, this shows that all eigenvalues of $\mathfrak{H}$, and thus all eigenvalues of $\mathcal{H}(w^*)$, are positive. □

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Appendix B Proofs of Lemmas 5.1 and 5.5

Proof of Lemma 5.1. We recall that in each of the cases (a)–(c), it has to be shown that the eigenvalues of L_{mix} are non-negative and that L_{mix} is real diagonalisable as $L_{\text{mix}} = X\Lambda X^{-1}$ with X of the specific form stated in each case.

Firstly, if L_{mix} is as in case (a), then L_{mix} is a symmetric real matrix and thus (by a standard result from linear algebra [31]) L_{mix} is orthogonally real diagonalisable. Furthermore, since W, P, B_γ^+, and B_γ^- are all symmetric matrices with non-negative entries, Lemmas 2.1 and 2.2 establish that L_W, Q_P, $L_{W_{\text{sym}}}$, $Q_{P_{\text{sym}}}$, $L_{B_\gamma^+}$, $Q_{B_\gamma^-}$, $L_{B_{\gamma\,\text{sym}}^+}$, and $Q_{B_{\gamma\,\text{sym}}^-}$ are all positive semidefinite with respect to the Euclidean inner product. Hence so is L_{mix} and thus its eigenvalues are non-negative.[62] This concludes the proof of part (a).

Secondly, if L_{mix} is as in case (b), then

$$L_{\text{mix}} = D_W^{-\frac{1}{2}} L_{W_{\text{sym}}} D_W^{\frac{1}{2}} + \gamma D_P^{-\frac{1}{2}} Q_{P_{\text{sym}}} D_P^{\frac{1}{2}} = D_W^{-\frac{1}{2}} \left(L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}} \right) D_W^{\frac{1}{2}},$$

where for the second equality we used the assumption that $D_P = D_W$. Hence L_{mix} is similar to the real diagonalisable matrix $L_{W_{\text{sym}}} + Q_{P_{\text{sym}}}$ and thus both matrices have the same eigenvalues and L_{mix} is real diagonalisable (but not orthogonally). Moreover, from Lemma 2.1 we know that $L_{W_{\text{rw}}}$ is positive semidefinite with respect to the W-degree-weighted inner product and from Lemma 2.2 we have that $Q_{P_{\text{rw}}}$ is positive semidefinite with respect to the P-degree-weighted inner product. Under the assumption $D_W = D_P$ these inner products are equal and thus L_{mix} is positive semidefinite with respect to this inner product. Hence its eigenvalues are non-negative.

Since, by definition of $\tilde{X}$,

$$\left(L_{W_{\text{sym}}} + Q_{W_{\text{sym}}} \right) \tilde{X} = \tilde{X}\Lambda,$$

we get

$$L_{\text{mix}} D_W^{-\frac{1}{2}} \tilde{X} = D_W^{-\frac{1}{2}} \left(L_{W_{\text{sym}}} + Q_{W_{\text{sym}}} \right) \tilde{X} = D_W^{-\frac{1}{2}} \tilde{X}\Lambda,$$

hence $X = D_W^{-\frac{1}{2}} \tilde{X}$. From case (a) we know that $\tilde{X}^{-1} = \tilde{X}^T$, and thus $X^{-1} = \tilde{X}^T D_W^{\frac{1}{2}}$.

Thirdly, let L_{mix} be as in case (c).

From (39) we recall that $L_{\text{mix}} = L_{B_{\gamma\,\text{rw}}^+} + D_{B_\gamma^+}^{-1} Q_{B_\gamma^-}$. Hence

$$D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}} = D_{B_\gamma^+}^{\frac{1}{2}} L_{B_{\gamma\,\text{rw}}^+} D_{B_\gamma^+}^{-\frac{1}{2}} + D_{B_\gamma^+}^{\frac{1}{2}} D_{B_\gamma^+}^{-1} Q_{B_\gamma^-} D_{B_\gamma^+}^{-\frac{1}{2}} = L_{B_{\gamma\,\text{sym}}^+} + D_{B_\gamma^+}^{-\frac{1}{2}} Q_{B_\gamma^-} D_{B_\gamma^+}^{-\frac{1}{2}}.$$

As the sum of two real symmetric matrices, the right-hand side above is a real symmetric matrix and thus has real eigenvalues. Since L_{mix} is similar to this matrix, it has the same, and thus also real, eigenvalues.

Moreover, by Lemma 2.2 part (a), we know that $Q_{B_\gamma^-}$ is positive semidefinite with respect to the Euclidean norm and thus so is $D_{B_\gamma^+}^{-\frac{1}{2}} Q_{B_\gamma^-} D_{B_\gamma^+}^{-\frac{1}{2}}$, since

$$\langle D_{B_\gamma^+}^{-\frac{1}{2}} Q_{B_\gamma^-} D_{B_\gamma^+}^{-\frac{1}{2}} u, u \rangle = \langle Q_{B_\gamma^-} D_{B_\gamma^+}^{-\frac{1}{2}} u, D_{B_\gamma^+}^{-\frac{1}{2}} u \rangle \geq 0.$$

By Lemma 2.1 part (b) also $L_{B_{\gamma\,\text{sym}}^+}$ is positive semidefinite with respect to the Euclidean inner product and thus so is the sum $L_{B_{\gamma\,\text{sym}}^+} + D_{B_\gamma^+}^{-\frac{1}{2}} Q_{B_\gamma^-} D_{B_\gamma^+}^{-\frac{1}{2}}$. It follows that this matrix, and thus also L_{mix}, have non-negative eigenvalues.[63] Finally, since $D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}}$ is real and symmetric, it is (real) orthogonally diagonalisable and thus $\tilde{X}^{-1} = \tilde{X}^T$, hence $X^{-1} = \tilde{X}^T D_{B_\gamma^+}^{\frac{1}{2}}$. □

[62] After all, if (λ, v) is a (real) eigenpair of L_{mix}, then $L_{\text{mix}} v = \lambda v$ and thus $\lambda \|v\|_2 = \langle \lambda v, v \rangle = \langle L_{\text{mix}} v, v \rangle \geq 0$.

[63] In particular, because $D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}}$ is positive semidefinite with respect to the Euclidean inner product, L_{mix} is positive semidefinite with respect to the B_γ^+-degree-weighted inner product:

$$\langle L_{\text{mix}} u, u \rangle_{B_\gamma^+} = \langle D_{B_\gamma^+}^{-1} D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} u, D_{B_\gamma^+}^{\frac{1}{2}} u \rangle_{B_\gamma^+} = \langle D_{B_\gamma^+}^{\frac{1}{2}} L_{\text{mix}} D_{B_\gamma^+}^{-\frac{1}{2}} (D_{B_\gamma^+}^{\frac{1}{2}} u), D_{B_\gamma^+}^{\frac{1}{2}} u \rangle \geq 0.$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Proof of Lemma 5.5. To prove positivity of the eigenvalues of L_{mix} it suffices to prove that L_{mix} is positive definite (see footnote 62). In each of the cases of the lemma, the assumptions of the corresponding part of Lemma 5.1 are satisfied. Hence, we already know that L_{mix} is positive semidefinite. It remains to prove that, for $u \in \mathcal{V}$, $\langle L_{\text{mix}}u, u\rangle = 0$ implies $u = 0$, where the inner product may be the Euclidean inner product, or one of the degree-weighted inner products. We treat each variant of L_{mix} separately: Cases I and II cover part (a); part (b) corresponds to Case III; and Cases IV–VI concern the variants from part (c). Let $u \in \mathcal{V}$.

Case I Let $L_{\text{mix}} = L_W + \gamma Q_P$ and assume the matrix P has at least one positive entry. Assume $\langle L_{\text{mix}}u, u\rangle = 0$. From (6) and (10) we have

$$0 = \langle L_{\text{mix}}u, u\rangle = \frac{1}{2}\sum_{i,j\in V}\left[\omega_{ij}(u_i - u_j)^2 + \gamma p_{ij}(u_i + u_j)^2\right].$$

Let $i, j \in V$, then

$$[\text{if } \omega_{ij} > 0, \text{ then } u_i = u_j] \quad \text{and} \quad [\text{if } p_{ij} > 0, \text{ then } u_i = -u_j]. \tag{64}$$

The first implication shows that u needs to have the same value on any two nodes that are connected by an edge. By induction, it follows that u needs to have the same value on any two nodes that are connected by a path in the graph. We recall that the graph based on W is assumed to be connected, hence any two nodes are connected by a path and thus u is constant on V.

By assumption there is a positive entry of P. If the entry is a diagonal entry p_{ii}, then (64) implies that $u_i = 0$. Since u is constant on V, $u = 0$. If the positive entry is an off-diagonal entry p_{ij} with $i \neq j$, then by (64) and the constancy of u, $u_i = -u_j = -u_i$. Again, we conclude that $u_i = 0$ and thus $u = 0$.

Case II Next, assume D_P is invertible (we recall D_W is invertible per assumption), let $L_{\text{mix}} = L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$. Since P is assumed to have non-negative entries, invertibility of D_P implies that the matrix P has at least one positive entry. Assume $\langle L_{\text{mix}}u, u\rangle = 0$. From (7) and (11) we have

$$0 = \langle L_{\text{mix}}u, u\rangle = \frac{1}{2}\sum_{i,j\in V}\left[\omega_{ij}\left(\frac{u_i}{\sqrt{(d_W)_i}} - \frac{u_j}{\sqrt{(d_W)_j}}\right)^2 + \gamma p_{ij}\left(\frac{u_i}{\sqrt{(d_P)_i}} + \frac{u_j}{\sqrt{(d_P)_j}}\right)^2\right].$$

Let $i, j \in V$, then

$$[\text{if } \omega_{ij} > 0, \text{ then } u_i = \sqrt{(d_W)_j^{-1}(d_W)_i}u_j] \quad \text{and} \quad [\text{if } p_{ij} > 0, \text{ then } u_i = -\sqrt{(d_P)_j^{-1}(d_P)_i}u_j]. \tag{65}$$

For $x \in \mathbb{R}$, we define the signum function $\operatorname{sgn}(x) = \begin{cases} 1, & \text{if } x > 0, \\ -1, & \text{if } x < 0, \\ 0, & \text{if } x = 0. \end{cases}$

Since D_W and D_P are non-negative invertible diagonal matrices, their diagonal entries are positive. Thus from (65) it follows that

$$[\text{if } \omega_{ij} > 0, \text{ then } \operatorname{sgn}(u_i) = \operatorname{sgn}(u_j)] \quad \text{and} \quad [\text{if } p_{ij} > 0, \text{ then } \operatorname{sgn}(u_i) = -\operatorname{sgn}(u_j)].$$

Thus, by a similar argument as in the previous case, connectedness of the graph based on W implies that $\operatorname{sgn}(u)$ is constant on V. Furthermore, based on P having a positive entry, we can use an argument as before to conclude that $\operatorname{sgn}(u) = 0$ and thus $u = 0$.

Case III Next, let $L_{\text{mix}} = L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$ and assume that $\langle L_{\text{mix}}u, u\rangle_W = 0$. Assume that the null model is such that $D_W = D_P$. This assumption allows us to use Lemma 5.1 part (b) to establish that L_{mix} is positive semidefinite. Moreover, it implies that the W-degree-weighted and P-degree-weighted inner products

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

are the same. Furthermore, since the diagonal elements of D_W are non-zero, so are the diagonal elements of D_P, and since P is a non-negative matrix, this implies that P has a positive entry. Because we also know, by (8) and (12), that

$$\langle L_{\text{mix}}u, u\rangle_W = \frac{1}{2}\sum_{i,j\in V}\left[\omega_{ij}(u_i-u_j)^2 + p_{ij}(u_i+u_j)^2\right],$$

the remainder of the proof in this case follows in exactly the same fashion as in case I.

In the remaining three cases, we always assume the conditions of part (c) of the lemma to be satisfied.

Case IV Let $L_{\text{mix}} = L_{B_\gamma^+} + Q_{B_\gamma^-}$ and assume that $\langle L_{\text{mix}}u, u\rangle = 0$. Repeating the argument from case I above with B_γ^+ instead of W and B_γ^- instead of P (and γ absent from the correct places), we find the analogue of (64):

$$[\text{if } (b_\gamma^+)_{ij} > 0, \text{ then } u_i = u_j] \quad \text{and} \quad [\text{if } (b_\gamma^-)_{ij} > 0, \text{ then } u_i = -u_j]. \tag{66}$$

If assumption (i) is satisfied, then the same argument as in the first case (with B_γ^+ and B_γ^- instead of W and P) proves $u = 0$.

If assumption (ii) holds, because the graph with adjacency matrix B_γ^- is connected, for all nodes $k, l \in V$ it holds that there exists a path in this graph connecting k and l and thus $u_k = \pm u_l$. Since this holds for all pairs of nodes, $|u|$ is constant on V. For the nodes i and j from the assumption we know that $u_i = u_j$ and, since they are connected by a path with an odd number of edges in the graph with adjacency matrix B_γ^-, we have $u_i = -u_j$. Thus $u_i = u_j$ and hence $|u| = 0$ and therefore $u = 0$.

If assumptions (iii) is satisfied, then for all $i \in V$, $(b_\gamma^-)_{ii} > 0$. Hence, for all $i \in V$, $u_i = -u_i$, thus $u_i = 0$ and $u = 0$.

Case V Next let $L_{\text{mix}} = L_{B_{\gamma\,\text{sym}}^+} + Q_{B_{\gamma\,\text{sym}}^-}$ and assume that $\langle L_{\text{mix}}u, u\rangle = 0$. We repeat the argument from case II with B_γ^+ instead of W and B_γ^- instead of P (and γ absent from the correct places) to find

$$[\text{if } (b_\gamma^+)_{ij} > 0, \text{ then } \operatorname{sgn}(u_i) = \operatorname{sgn}(u_j)] \quad \text{and} \quad [\text{if } (b_\gamma^-)_{ij} > 0, \text{ then } \operatorname{sgn}(u_i) = -\operatorname{sgn}(u_j)].$$

For each of the three assumptions (i), (ii), and (iii) we repeat the arguments from case IV, but for $\operatorname{sgn}(u)$ instead of u. Then we find under each of the assumptions that $\operatorname{sgn}(u) = 0$ and thus $u = 0$.

Case VI Finally, let $L_{\text{mix}} = L_{B_{\gamma\,\text{rw}}^+} + Q_{B_{\gamma\,\text{rw}}^-} - D_{B_\gamma^+}^{-1} D_B Q_{B_{\gamma\,\text{rw}}^-}$ and assume that $\langle L_{\text{mix}}u, u\rangle_{B_\gamma^+} = 0$. Then we use (39), (8), and (10) to compute

$$\begin{aligned} 0 = \langle L_{\text{mix}}u, u\rangle_{B_\gamma^+} &= \langle L_{B_{\gamma\,\text{rw}}^+}u, u\rangle_{B_\gamma^+} + \langle D_{B_\gamma^+}^{-1}Q_{B_\gamma^-}u, u\rangle_{B_\gamma^+} = \langle L_{B_{\gamma\,\text{rw}}^+}u, u\rangle_{B_\gamma^+} + \langle Q_{B_\gamma^-}u, u\rangle \\ &= \frac{1}{2}\sum_{i,j\in V}(b_\gamma^+)_{ij}(u_i-u_j)^2 + \frac{1}{2}\sum_{i,j\in V}(b_\gamma^-)_{ij}(u_i+u_j)^2. \end{aligned}$$

Thus, again we recover (66). From here the proof proceeds in the same way as in case IV and we conclude that, under each of the three assumptions (i), (ii), and (iii), $u = 0$. □

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Appendix C Proof of Lemma 6.1

Proof of Lemma 6.1.

(a) Since the infinity operator norm is sub-multiplicative[64] and $U(\tau) = e^{-\tau L_{\text{mix}}} U^0$, we get

$$\|U(\tau) - U^0\|_\infty \le \|e^{-\tau L_{\text{mix}}} - I\|_\infty \|U^0\|_\infty \le \|U^0\|_\infty \sum_{l=1}^{\infty} \frac{\tau^l}{l!} \|L_{\text{mix}}\|_\infty^l = \|U^0\|_\infty \left(e^{\tau \|L_{\text{mix}}\|_\infty} - 1\right). \quad (67)$$

Since $U^0 \in Pt(K)$, we have $\|U^0\|_\infty = K$ and the first result follows. Let δ_{ij} denote the Kronecker delta. We recall that W, P, B_γ^+, and B_γ^- have non-negative entries and that in the cases in which expressions of the form x^{-1} or $x^{-\frac{1}{2}}$ appear, x is assumed to be positive (as is needed to have L_{mix} be well-defined in those cases).

(i) From $(L_{\text{mix}})_{ij} = (d_W)_i \delta_{ij} - \omega_{ij} + \gamma (d_P)_i \delta_{ij} + \gamma p_{ij}$, it follows that

$$\begin{aligned} \|L_{\text{mix}}\|_\infty &\le \max_{i \in V} \left(\sum_{j \in V} (d_W)_i \delta_{ij} + \sum_{j \in V} \omega_{ij} + \sum_{j \in V} \gamma (d_P)_i \delta_{ij} + \sum_{j \in V} \gamma p_{ij} \right) \\ &= \max_{i \in V} \left((d_W)_i + (d_W)_i + \gamma (d_P)_i + \gamma (d_P)_i \right) \le 2 \max_{i \in V} (d_W)_i + 2 \max_{i \in V} (d_P)_i \\ &= L^{\max}. \end{aligned}$$

(ii) Since $(L_{\text{mix}})_{ij} = \delta_{ij} - (d_W)_i^{-\frac{1}{2}} \omega_{ij} (d_W)_j^{-\frac{1}{2}} + \gamma \delta_{ij} + (d_P)_i^{-\frac{1}{2}} \gamma p_{ij} (d_P)_j^{-\frac{1}{2}}$, we obtain

$$\|L_{\text{mix}}\|_\infty \le \max_{i \in V} \left(1 + \frac{(d_W)_i^{\frac{1}{2}}}{d_{W,\min}^{\frac{1}{2}}} + \gamma + \gamma \frac{(d_P)_i^{\frac{1}{2}}}{d_{P,\min}^{\frac{1}{2}}} \right) \le 1 + \gamma + \max_{i \in V} \frac{(d_W)_i^{\frac{1}{2}}}{d_{W,\min}^{\frac{1}{2}}} + \gamma \max_{i \in V} \frac{(d_P)_i^{\frac{1}{2}}}{d_{P,\min}^{\frac{1}{2}}} = L^{\max}.$$

(iii) From $(L_{\text{mix}})_{ij} = \delta_{ij} - (d_W)_i^{-1} \omega_{ij} + \gamma \delta_{ij} + \gamma (d_P)_i^{-1} p_{ij}$, we get

$$\|L_{\text{mix}}\|_\infty \le \max_{i \in V} (1 + 1 + \gamma + \gamma) = L^{\max}.$$

(iv) Using $(L_{\text{mix}})_{ij} = (d_{B_\gamma^+})_i \delta_{ij} - (b_\gamma^+)_{ij} + (d_{B_\gamma^-})_i \delta_{ij} + (b_\gamma^-)_{ij}$ yields

$$\|L_{\text{mix}}\|_\infty \le \max_{i \in V} \left(2(d_{B_\gamma^+})_i + 2(d_{B_\gamma^-})_i \right) \le 2 \max_{i \in V} (d_{B_\gamma^+})_i + 2 \max_{i \in V} (d_{B_\gamma^-})_i = L^{\max}.$$

(v) Since $(L_{\text{mix}})_{ij} = \delta_{ij} - (d_{B_\gamma^+})_i^{-\frac{1}{2}} (b_\gamma^+)_{ij} (d_{B_\gamma^+})_j^{-\frac{1}{2}} + \delta_{ij} + (d_{B_\gamma^-})_i^{-\frac{1}{2}} (b_\gamma^-)_{ij} (d_{B_\gamma^-})_j^{-\frac{1}{2}}$, we find

$$\|L_{\text{mix}}\|_\infty \le \max_{i \in V} \left(2 + \frac{(d_{B_\gamma^+})_i^{\frac{1}{2}}}{d_{B_\gamma^+,\min}^{\frac{1}{2}}} + \frac{(d_{B_\gamma^-})_i^{\frac{1}{2}}}{d_{B_\gamma^-,\min}^{\frac{1}{2}}} \right) \le 2 + \max_{i \in V} \frac{(d_{B_\gamma^+})_i^{\frac{1}{2}}}{d_{B_\gamma^+,\min}^{\frac{1}{2}}} + \max_{i \in V} \frac{(d_{B_\gamma^-})_i^{\frac{1}{2}}}{d_{B_\gamma^-,\min}^{\frac{1}{2}}} = L^{\max}.$$

(vi) Since $(L_{\text{mix}})_{ij} = \delta_{ij} - (d_{B_\gamma^+})_i^{-1} (b_\gamma^+)_{ij} + (d_{B_\gamma^+})_i^{-1} \left((d_{B_\gamma^-})_i + (b_\gamma^-)_{ij} \right)$ by (39), we obtain

$$\|L_{\text{mix}}\|_\infty \le \max_{i \in V} \left(2 + 2 \frac{(d_{B_\gamma^-})_i}{(d_{B_\gamma^+})_i} \right) \le 2 \left(1 + \max_{i \in V} \frac{(d_{B_\gamma^-})_i}{d_{B_\gamma^+,\min}} \right) = L^{\max}.$$

(b) Let $K = 2$. Each column of U satisfies the same ODE:

$$\frac{dU_{*1}}{dt} = -L_{\text{mix}} U_{*1} \quad \text{and} \quad \frac{dU_{*2}}{dt} = -L_{\text{mix}} U_{*2}.$$

[64] If $A \in \mathbb{R}^{n \times p}$ and $B \in \mathbb{R}^{p \times q}$, then

$$\begin{aligned} \|AB\|_\infty &= \max_{i \in \{1,\dots,n\}} \sum_{j=1}^{q} |(AB)_{ij}| = \max_{i \in \{1,\dots,n\}} \sum_{j=1}^{q} \left| \sum_{l=1}^{p} A_{il} B_{lj} \right| \le \max_{i \in \{1,\dots,n\}} \sum_{j=1}^{q} \sum_{l=1}^{p} |A_{il}| |B_{lj}| \\ &\le \max_{\substack{i \in \{1,\dots,n\} \\ k \in \{1,\dots,p\}}} \sum_{j=1}^{q} \sum_{l=1}^{p} |A_{il}| |B_{kj}| = \|A\|_\infty \|B\|_\infty. \end{aligned}$$

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Setting $v := \frac{1}{2}(U_{*1} - U_{*2})$, we obtain $\frac{dv}{dt} = -L_{\text{mix}} v$ with initial condition $v(0) = \frac{1}{2}(U^0_{*1} - U^0_{*2})$. Because $K = 2$, each entry of the vector $v(0)$ is either 1 or -1. Thus, if for an $i \in V$, $|v_i(\tau) - v_i(0)| < 1$, then $\operatorname{sgn}(v_i(\tau)) = \operatorname{sgn}(v_i(0))$ and thus $U_{i1}(\tau) > U_{i2}(\tau)$ if and only if $U_{i1}(0) > U_{i2}(0)$, and $U_{i1}(\tau) < U_{i2}(\tau)$ if and only if $U_{i1}(0) < U_{i2}(0)$. Thus, if for all $i \in V$, $|v_i(\tau) - v_i(0)| < 1$, then $U^1 = U^0$. If $\tau < \tau_{\text{low}}$, it follows by (67) applied to the column vector $v - v(0)$ instead of the matrix $U - U^0$ (with $\|v(0)\|_\infty = 1$) that, $\|v(\tau) - v(0)\|_\infty < 1$ and thus, for all $i \in V$, $|v_i(\tau) - v_i(0)| < 1$.

(c) In the notation of Lemma 5.1, we have $L_{\text{mix}} = X \Lambda X^{-1}$. By a property of matrix exponentials, we find that $e^{-\tau L_{\text{mix}}} = X e^{-\tau \Lambda} X^{-1}$. Lemma 5.1 shows that for each choice of L_{mix}, $X^{-1} = \tilde{X} D^{\frac{1}{2}}$ for some diagonal and invertible matrix D with positive diagonal entries and for an orthogonal matrix $\tilde{X}$. Hence

$$\begin{aligned}
\left(e^{-\tau L_{\text{mix}}}\right)^T C e^{-\tau L_{\text{mix}}} &= \left(X e^{-\tau \Lambda} X^{-1}\right)^T C X e^{-\tau \Lambda} X^{-1} \\
&= \left(D^{-\frac{1}{2}} \tilde{X}^{-1} e^{-\tau \Lambda} \tilde{X} D^{\frac{1}{2}}\right)^T C D^{-\frac{1}{2}} \tilde{X}^{-1} e^{-\tau \Lambda} \tilde{X} D^{\frac{1}{2}} \\
&= D^{\frac{1}{2}} \tilde{X}^{-1} e^{-\tau \Lambda} \tilde{X} D^{-\frac{1}{2}} C D^{-\frac{1}{2}} \tilde{X}^{-1} e^{-\tau \Lambda} \tilde{X} D^{-\frac{1}{2}} \\
&= D^{\frac{1}{2}} \tilde{X}^{-1} e^{-2\tau \Lambda} \tilde{X} D^{-\frac{1}{2}},
\end{aligned}$$

where the last equality follows if $C = D$. From Lemma 5.1 we see that $D = D_W$ if $L_{\text{mix}} = L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$, $D = D_{B^+_\gamma}$ if $L_{\text{mix}} = L_{B^+_{\gamma\,\text{rw}}} + Q_{B^-_{\gamma\,\text{rw}}} - D^{-1}_{B^+_\gamma} D_{B_\gamma} Q_{B^-_{\gamma\,\text{rw}}}$, and $D = I$ for the other choices of L_{mix}. Thus with the choice of C as stated in the current lemma, we have

$$\left\|e^{-\tau L_{\text{mix}}}\right\|^2_{\text{Fr},C} = \operatorname{tr}\left(D^{\frac{1}{2}} \tilde{X}^{-1} e^{-2\tau \Lambda} \tilde{X} D^{-\frac{1}{2}}\right) = \operatorname{tr}\left(\tilde{X}^{-1} e^{-2\tau \Lambda} \tilde{X}\right) = \operatorname{tr}\left(e^{-2\tau \Lambda}\right) = \left\|e^{-\tau \Lambda}\right\|^2_{\text{Fr}},$$

where we used twice the cyclic property of the trace. Since the trace of a square matrix is the sum of its eigenvalues, we find

$$\left\|e^{-\tau L_{\text{mix}}}\right\|_{\text{Fr},C} = \left\|e^{-\tau \Lambda}\right\|_{\text{Fr}} \le e^{-\tau \lambda_1}.$$

By property (4) of the C-Frobenius norm, we conclude that

$$\|U(\tau)\|_{\text{Fr},C} = \left\|e^{-\tau L_{\text{mix}}} U^0\right\|_{\text{Fr},C} \le \left\|e^{-\tau L_{\text{mix}}}\right\|_{\text{Fr},C} \left\|U^0\right\|_{\text{Fr}} \le e^{-\tau \lambda_1} \left\|U^0\right\|_{\text{Fr}}.$$

(d) Since C is diagonal and has positive diagonal entries, we compute

$$\begin{aligned}
\|U(\tau)\|_{\text{Fr},C} &= \sqrt{\sum_{i \in V} \sum_{k=1}^{K} C_{ii} |U_{ik}(\tau)|^2} \ge c_{\min}^{\frac{1}{2}} \sqrt{\sum_{i \in V} \sum_{k=1}^{K} |U_{ik}(\tau)|^2} \ge c_{\min}^{\frac{1}{2}} \sqrt{\max_{i \in V} \sum_{k=1}^{K} |U_{ik}(\tau)|^2} \\
&\ge c_{\min}^{\frac{1}{2}} K^{-\frac{1}{2}} \max_{i \in V} \left(\sum_{k=1}^{K} |U_{ik}(\tau)|\right) = c_{\min}^{\frac{1}{2}} K^{-\frac{1}{2}} \|U(\tau)\|_\infty.
\end{aligned}$$

The third inequality follows from the Cauchy–Schwarz inequality for the Euclidean inner product and norm applied to the vector $\max_{i \in V} |U_{i*}(\tau)| \in \mathbb{R}^K$ and the vector of ones in $\mathbb{R}^K$. Using the result of part (c), we thus have

$$\|U(\tau)\|_\infty \le K^{\frac{1}{2}} c_{\min}^{-\frac{1}{2}} \|U(\tau)\|_{\text{Fr},C} \le K^{\frac{1}{2}} c_{\min}^{-\frac{1}{2}} e^{-\tau \lambda_1} \left\|U^0\right\|_{\text{Fr}}.$$

Hence, if $\tau > \tau_{\text{upp}}$, then $\|U(\tau)\|_\infty < \theta$.

□

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press

Appendix D Weyl's inequality and rank–one matrix updates

If $A \in \mathbb{C}^{n\times n}$ is a Hermitian matrix, it is known that it has n real eigenvalues $\lambda_i(A)$ (counted according to algebraic multiplicity), which we label according to their ordering:

$$\lambda_1(A) \leq \lambda_2(A) \leq \ldots \leq \lambda_n(A).$$

Weyl's inequality (which we present below without proof) gives a bound on the eigenvalues of the sum of Hermitian matrices. That result is followed by the theorem of rank–one matrix updates (also without proof), which illustrates why the eigenvalues of the original matrix and the rank–one update matrix are interleaved.

Theorem D.1. *(Weyl's inequality [73]) Let $A, B \in \mathbb{C}^{n\times n}$ be Hermitian matrices. Then, for all $i \in \{1, \ldots, n\}$,*

$$\lambda_i(A) + \lambda_1(B) \leq \lambda_i(A + B) \leq \lambda_i(A) + \lambda_n(B).$$

Theorem D.2. *(Rank–one matrix updates [74, 75]) If $A, B \in \mathbb{C}^{n\times n}$ are positive semidefinite Hermitian matrices and B has rank at most equal to one, then, for all $i \in \{1, \ldots, n-1\}$,*

$$\lambda_i(A + B) \leq \lambda_{i+1}(A) \leq \lambda_{i+1}(A + B).$$

Proof. This follows from [69, Theorem 1]. □

Corollary D.3. *Let $L_{mix} = L_{W_{sym}} + \gamma Q_{P_{sym}}$, where $P = P^{NG}$ is obtained from the Newman–Girvan null model. Then, for all $i \in \{1, \ldots, n\}$,*

$$\lambda_i(L_{W_{sym}}) + \gamma \leq \lambda_i(L_{mix}) \leq \lambda_i(L_{W_{sym}}) + 2\gamma. \tag{68}$$

Moreover, for all $i \in \{1, \ldots, n-1\}$,

$$\lambda_i(L_{mix}) \leq \lambda_{i+1}(L_{W_{sym}} + \gamma I) = \lambda_{i+1}(L_{W_{sym}}) + \gamma \leq \lambda_{i+1}(L_{mix}). \tag{69}$$

Finally, if $L_{mix} = L_{W_{rw}} + \gamma Q_{P_{rw}}$ instead, then (68) and (69) also hold, both in the original form and with $L_{W_{rw}}$ replacing $L_{W_{sym}}$.

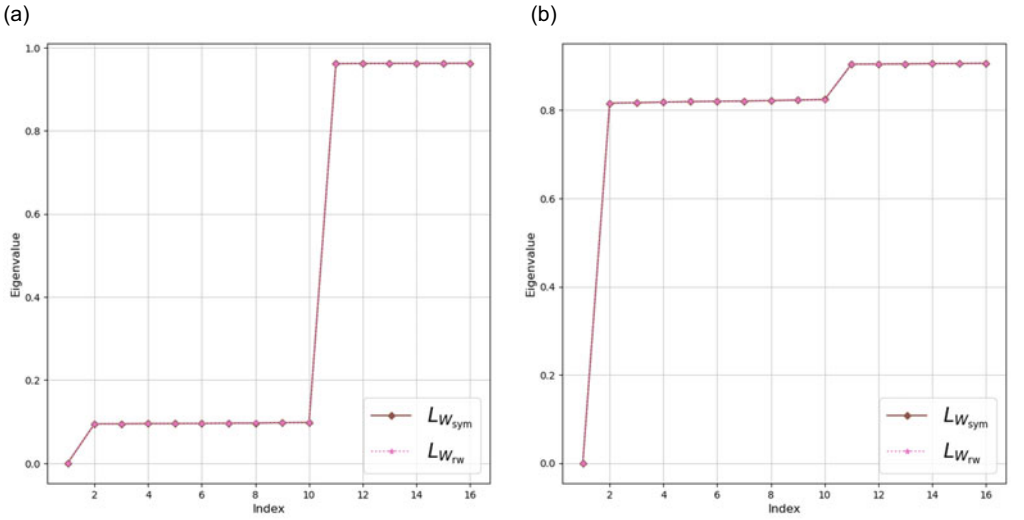
Proof. Since $D_P = D_W$, the matrix $D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}}$ can be written as zz^T, where the column vector $z \in \mathbb{R}^{|V|}$ has entries $z_i = (\mathrm{vol}_W(V))^{-\frac{1}{2}} (d_W)_i^{\frac{1}{2}}$. Thus $D_P^{-\frac{1}{2}} P D_P^{-\frac{1}{2}}$ has rank one and hence all its eigenvalues but one are equal to zero. The only non-zero eigenvalue equals one. (It can be checked that the vector v with entries $v_i := (d_W)_i^{\frac{1}{2}}$ is a corresponding eigenvector.) Thus $Q_{P_{\mathrm{sym}}} = I + zz^T$ has one eigenvalue equal to 2 and $|V| - 1$ eigenvalues equal to 1. In particular $\lambda_1(Q_{P_{\mathrm{sym}}}) = 1$ and $\lambda_{|V|}(Q_{P_{\mathrm{sym}}}) = 2$. Because both $L_{W_{\mathrm{sym}}}$ and $\gamma Q_{P_{\mathrm{sym}}}$ are real symmetric matrices, from Theorem D.1 it follows that, for all $i \in \{1, \ldots, n\}$, (68) holds.

Since zz^T has non-negative eigenvalues, it is positive semidefinite. By Lemma 2.1 also $L_{W_{\mathrm{sym}}} + \gamma I$ is positive semidefinite. Since $L_{\mathrm{mix}} = L_{W_{\mathrm{sym}}} + \gamma I + \gamma zz^T$, it follows from Theorem D.2 that, for all $i \in \{1, \ldots, n-1\}$, (69) holds.

Since $L_{W_{\mathrm{sym}}}$ and $L_{W_{\mathrm{rw}}}$ have the same eigenvalues and, by Remark 5.2, also $L_{W_{\mathrm{sym}}} + \gamma Q_{P_{\mathrm{sym}}}$ and $L_{W_{\mathrm{rw}}} + \gamma Q_{P_{\mathrm{rw}}}$ have the same eigenvalues, the final statement of this corollary follows immediately. □

Remark D.4. It can be observed in Figure 5 that there is a jump between the 9th and 10th eigenvalues of the operators included in the plot, in the SBM example from Section 7.3 with 10 blocks. For the SBM with strong community structure, this jump is more pronounced than for the SBM with weak community structure. A jump in the spectrum of the graph Laplacian for a graph with a strong community structure with K communities after the K^{th} eigenvalue is expected and in fact a key reason why graph Laplacians are useful for clustering; see for example [29, Section 4]. For the SBM this is confirmed by the plot in Figure 9a; in Figure 9b we see that also for the SBM with the weak community structure a jump occurs after the 10th eigenvalues, but a much smaller one.

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press



SBM with the strong community structure. SBM with the weak community structure.

Figure 9. *SBM with strong and weak community structure (see Section 7.3 for details): spectra of $L_{W_{sym}}$ and $L_{W_{rw}}$. As expected [74], both operators have the same eigenvalues.*

Let us consider the operator $L_{\text{mix}} = L_{W_{\text{sym}}} + \gamma Q_{P_{\text{sym}}}$ which is included in the plots of Figure 5. Applying (69) yields

$$\lambda_9(L_{\text{mix}}) \leq \lambda_{10}(L_{W_{\text{sym}}}) + \gamma \leq \lambda_{10}(L_{\text{mix}}) \leq \lambda_{11}(L_{W_{\text{sym}}}) + \gamma \leq \lambda_{11}(L_{\text{mix}}).$$

Hence, given the jump between the 10^{th} and 11^{th} eigenvalue of $L_{W_{\text{sym}}}$, there are three principal scenarios:

1. there is a similarly large jump between $\lambda_9(L_{\text{mix}})$ and $\lambda_{10}(L_{\text{mix}})$,
2. or there is a similarly large jump between $\lambda_{10}(L_{\text{mix}})$ and $\lambda_{11}(L_{\text{mix}})$,
3. or there are two smaller jumps between $\lambda_9(L_{\text{mix}})$ and $\lambda_{10}(L_{\text{mix}})$ and between $\lambda_{10}(L_{\text{mix}})$ and $\lambda_{11}(L_{\text{mix}})$.

In Figure 5 one encounters the first scenario.

By the last statement of Corollary D.3, we can argue similarly for the eigenvalues of $L_{W_{\text{rw}}}$ and $L_{\text{mix}} = L_{W_{\text{rw}}} + \gamma Q_{P_{\text{rw}}}$.

Cite this article: Li Z., van Gennip Y. and John V. (2026). An MBO method for modularity optimisation based on total variation and signless total variation. *European Journal of Applied Mathematics*, **37**, 743–825. https://doi.org/10.1017/S095679252400072X

https://doi.org/10.1017/S095679252400072X Published online by Cambridge University Press