Multidimensional Hopfield Networks for clustering

Gergely Stomfai
University of Cambridge, Synerise

Łukasz Sienkiewicz
Synerise

Barbara Rychalska
Synerise

Abstract

We present the Multidimensional Hopfield Network (DHN), a natural generalisation of the Hopfield Network. In our theoretical investigations we focus on DHNs with a certain activation function and provide energy functions for them. We conclude that these DHNs are convergent in finite time, and are equivalent to greedy methods that aim to find graph clusterings of locally minimal cuts. We also show that the general framework of DHNs encapsulates several previously known algorithms used for generating graph embeddings and clusterings. Namely, the Cleora graph embedding algorithm, the Louvain method, and the Newman's method can be cast as DHNs with appropriate activation function and update rule. Motivated by these findings we provide a generalisation of Newman's method to the multidimensional case.

1 Multidimensional Hopfield Networks

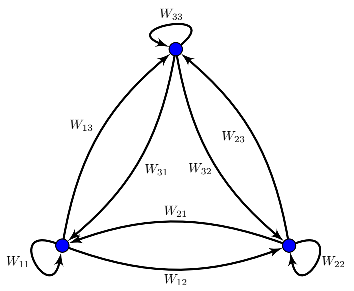
For the reader's convenience we introduce some notation that is used throughout the paper. If n, d are positive integers, then $\mathbb{M}_{n\times d}(\mathbb{R})$ denotes the set of $n \times d$ matrices with entries in $\mathbb{R}$. If $X \in \mathbb{M}_{n\times d}(\mathbb{R})$ is a matrix, then X^T denotes its transpose and provided that $n = d$, the trace of X is denoted by $\mathrm{Tr}(X)$. Let $F : A \to B$ be a function between the sets A, B. Then its range i.e. the subset

$$\big\{F(a) \,\big|\, a \in A\big\} \subseteq B$$

is denoted by $\mathrm{ran}(F)$. Now we turn to the general framework that encapsulates all the methods to follow.

Definition 1.1. Let n, d be positive integers. *A d-dimensional Hopfield network with n neurons* is a triple (W, B, F) such that $W \in \mathrm{M}_{n\times n}(\mathbb{R})$, $B \in \mathrm{M}_{n\times d}(\mathbb{R})$ and $F : \mathbb{R}^d \to \mathbb{R}^d$ is a function, called the *activation function*. Further, each neuron i of the DHN has a *state vector* $X_i \in \mathbb{R}^d$.

Let us interpret the notion above in a more intuitive way, which is more familiar to machine learning practitioners. Suppose that (W, B, F) is a two-dimensional Hopfield network with three neurons. Then $W \in \mathrm{M}_{3\times 3}(\mathbb{R})$ is interpreted as the connection weights matrix between the three neurons.



The entries of the matrix $B \in \mathrm{M}_{3\times 2}(\mathbb{R})$ are biases corresponding to each neuron and each state, while $F : \mathbb{R}^2 \to \mathbb{R}^2$ is the common activation function for all the neurons in the network. The state

Associative Memory & Hopfield Networks in 2023. NeurIPS 2023 workshop.

vectors of the neurons are vectors in $\mathbb{R}^2$. From this understanding of the network it is clear, that DHNs are a subclass of recurrent neural networks.
Since recurrent neural networks determine discrete time dynamical systems on the space of neuron states, each DHN gives rise to such a dynamical system.

Definition 1.2. Let (W, B, F) be a d-dimensional Hopfield network with n-neurons. Let $X_i(t) \in \mathbb{R}^d$ denote the state of the i-th neuron at time $t \in \mathbb{N}$. We consider the following two types of discrete time dynamical systems determined by (W, B, F) and the initial states $X_i(0)$ of the neurons.

- In *serial mode of operation* given that the neurons are in the states
$$X_1(t), ..., X_n(t) \in \mathbb{R}^d$$
at time $t \in \mathbb{N}$, we pick a neuron $i \in \{1, ..., n\}$ and update it's state according to the rule
$$X_i(t+1) = F\left(\sum_{j=1}^{n} W_{ij} \cdot X_j(t) + B_i\right)$$
while the states of all other neurons stay unchanged, i.e. $X_j(t+1) = X_j(t)$ for all $j \neq i$. Note that the dynamics of the system depends on the order in which the neurons are updated.

- In *parallel mode of operation* given that the neurons are in the states
$$X_1(t), ..., X_n(t) \in \mathbb{R}^d$$
at time $t \in \mathbb{N}$, all neurons are updated simultaneously by the same rule as in serial mode (see Remark 1.4).

Remark 1.3. Let (W, B, F) be a d-dimensional Hopfield network with n-neurons operating in either serial or parallel mode. In the sequel we denote by $X(t)$ the matrix which has the neuron state vectors $X_1(t), ..., X_n(t)$ as rows and call it the *matrix of neuron states*.

One can find both one and two dimensional DHNs that mimic the behaviour of the original Hopfield networks introduced in Hopfield (1982).
In some cases, DHNs are only updated according to the parallel mode. This comes with some computational ease, and we can also be a bit more vague about the nature of the activation function, as Remark 1.5 explains.

Remark 1.4. If $F : \mathbb{R}^d \to \mathbb{R}^d$, then for a $M \in \mathrm{M}_{n\times d}(\mathbb{R})$ we denote by $F(M)$ the result of applying F to every row of M. Now let (W, B, F) be a d-dimensional Hopfield network with n neurons. Then, in parallel mode of operation the update can be written in the following form
$$X(t+1) = F\left(WX(t) + B\right)$$
for every $t \in \mathbb{N}$.

Remark 1.5. The matrix form of the parallel update noted in Remark 1.4 shows that for this mode of operation one can set F to be a function on matrices, i.e. $F : \mathbb{M}_{n\times d}(\mathbb{R}) \to \mathbb{M}_{n\times d}(\mathbb{R})$. Although strictly speaking Definition 1.1 does not allow this, as long as the network is only updated in parallel mode, it poses no practical difficulty, but allows more intricate behaviour in some cases and will prove to be useful later.

Example 1.6. Consider a weighted graph with n nodes and edge weight matrix W. Then for a fixed positive integer d we set $F : \mathbb{R}^d \to \mathbb{R}^d$ to be the l_2-normalization function, i.e.
$$F(x) = \frac{x}{\|x\|_2}$$
if $x \neq 0$ and $F(0) = 0$. This gives rise to a d-dimensional Hopfield network $(W, 0, F)$ with n neurons. Further, if the entries of the initial neuron states matrix $X(0) \in \mathbb{M}_{n\times d}(\mathbb{R})$ are sampled from $\mathcal{U}(-1, 1)$, then $(W, 0, F)$ operating in parallel mode is equivalent to running the Cleora algorithm from Rychalska et al. (2021).

Remark 1.7. Let (W, B, F) be a d-dimensional Hopfield network with n-neurons operating in either serial or parallel mode. Then by means of Remark 1.4 and without loss of generality we may assume that $X(t) \in \mathrm{ran}(F)$ for every $t \in \mathbb{N}$ where
$$\mathrm{ran}(F) = \left\{F(M) \,\middle|\, M \in \mathbb{M}_{n\times d}(\mathbb{R})\right\}$$

2 DHNs with classification function

In this section we investigate the convergence properties of DHNs with a particular activation function.

Definition 2.1. The function $\mathrm{cl} : \mathbb{R}^d \to \mathbb{R}^d$ defined as

$$\mathrm{cl}(x)_i = \delta_i \left(\underset{j}{\operatorname{argmax}} \, x_j \right)$$

for each $i \in \{1, ..., d\}$ is the *classification function for* $\mathbb{R}^d$. We also denote the image of cl in $\mathbb{R}^d$ by $\mathbb{L}_d$ and call it *the label space of dimension* d.

For DHNs with classification function as the activation we can provide an energy function – a function that is decreasing along the trajectories obtained by serial mode operations.

Theorem 2.2. *Suppose that* $W \in \mathbb{M}_{n\times n}(\mathbb{R})$ *is a symmetric matrix with nonnegative entries on the diagonal and* $B \in \mathbb{M}_{n\times d}(\mathbb{R})$*. Then the function*

$$V(X) = -Tr\left(X^T W X + 2X^T B\right)$$

is an energy function function for the multidimensional Hopfield network (W, B, cl) *operating in serial mode.*

Theorem 2.3. *Let* (W, B, cl) *be a* d*-dimensional Hopfield network with* n *neurons. Suppose that* W *is symmetric. Then the following assertions hold.*

(1) *If* W *has nonnegative entries on the diagonal and* (W, B, cl) *operates in a serial mode, then the corresponding dynamical system converges to a stable state for every initial state of the neurons.*

(2) *If* (W, B, cl) *is operating in parallel mode, then the corresponding dynamical system converges to a cycle of length at most* 2 *for every initial state of the neurons.*

For proofs we refer the reader to Appendix A.1. Moreover, the results of Bruck (1990) concerning the relationship between Hopfield networks and greedy algorithms solving graph min-cut problem can also be generalised to DHNs with classification function. DHNs though, solve a generalised version of min-cut problem. We discuss this at length in Appendix A.2.

3 DHNs for optimising modularity

The modularity matrix of a graph was defined in Newman (2006b). We exhibit that two well known methods of maximising graph modularity, namely Newman's method from Newman (2006b) and the Louvain method from Blondel et al. (2008) can be viewed as certain DHNs. Extended discussion on the topic, like proofs etc. can be found in Appendix A.3

3.1 The Louvain method

The Louvain method Blondel et al. (2008) is a popular graph clustering method, based around the heuristic idea of greedy local search, using the modularity of the clustering as an objective function to maximise. Strictly speaking, after reaching a local optimum, the Louvain method merges nodes in the same cluster together – this part of the algorithm is irrelevant for our analysis.

Proposition 3.1. *There exists a large class of weighted graphs such that if G is in the class, then there exists a DHN with classification function as activation such that running the Louvain method on G with any initial choice of clusters can be cast as running this DHN in serial mode of operation and some initialization.*

For the proof see Theorem A.14 and Proposition A.15 in A.3.

3.2 Newman's and other iterative methods

Similarly one can easily construct a DHN which is equivalent to the power method used in finding the leading eigenvector of matrices. Using this observation one can also implement Newman's method from Newman (2006b). For details see A.3.

	$F = P_{V_d(\mathbb{R}^n)}$	$F = \mathrm{cl}$
serial	SGNM	LMS
parallel	GNM	PLMS

Table 1: Propagation methods naturally arising from generalising Newman's method to the multidimensional case. The abbreviations refer to: SGNM serial generalized Newman method, LMS: Louvain method search, which refers to the first phase of the Louvain method, PLMS: parallel Louvain method search, GNM: Generalised Newman method.

	Cora	Citeseer	PubMed	Photos
LMS	0.5510	0.6659	0.5602	**0.6898**
GNM	0.5754	0.5901	0.4489	0.4901
GNM + one iteration LMS	**0.7147**	**0.7292**	**0.6580**	0.6786
PLMS	0.5019	0.5292	0.4245	0.6042
SGNM	0.4267	0.4587	0.2844	0.0621

Table 2: Comparison of modularity value for some of the methods from Table 1. All parallel methods were halted when converged. Both parallel methods were run using 64 dimensional neuron states, and thus were limited to 64 clusters at any point in time. SGNM was run for 3 iterations.

Writing Newman's method as a DHN allows us to extend it to the case of multi-label clustering, simply by considering a DHN of higher dimensions. Let n, d be positive integers. Then a subset

$$V_d(\mathbb{R}^n) = \left\{O \in \mathbb{M}_{n\times d}(\mathbb{R}) \,\middle|\, O^T O = I_d\right\} \subseteq \mathbb{M}_{n\times d}(\mathbb{R})$$

is *the Stiefel manifold of orthonormal d-frames in* $\mathbb{R}^n$. Let $P_{V_d(\mathbb{R}^n)} : \mathbb{M}_{n\times d}(\mathbb{R}) \to V_d(\mathbb{R}^n)$ be such that $P_{V_d(\mathbb{R}^n)}(M) = \mathrm{argmax}_{S\in V_d(\mathbb{R}^n)} \operatorname{Tr}\left(S^T M\right)$ for every $M \in \mathbb{M}_{n\times d}(\mathbb{R})$. We set $F = P_{V_d(\mathbb{R}^n)}$ - thus ensuring that $X(t) \in V_d(\mathbb{R}^n)$. See Remark 1.5 about using matrix valued functions for F. The existence of $P_{V_d(\mathbb{R}^n)}$ can be verified constructively, for example by considering the QR decomposition algorithm (see Allaire and Kaber (2008) for more details).

Listing 1: Generalised Newman method.

```
input modularity matrix $Q$
initialise neuron states matrix of $(Q, \mathbf{0}, P_{V_d(\mathbb{R}^n)})$ randomly
until the neuron states matrix of $(Q, \mathbf{0}, P_{V_d(\mathbb{R}^n)})$ converges
    update neurons of $(Q, \mathbf{0}, P_{V_d(\mathbb{R}^n)})$ in parallel mode
return $\mathrm{cl}$(neuron states matrix)
```

By varying the mode of operation and the post-processing function we obtain several new methods that can again be used to find clusterings. We summarise some of these in Table 1. The methods GNM, SGNM and PLMS are previously unexplored according to our knowledge. They are methods that inherit properties of both the Louvain and Newman's methods to some extent, and therefore might be of interest for finding composite methods combining the strengths of the previously existing ones. We present some experimental result obtained from running these methods on well-known graphs in Table 2.

We emphasize that the settings we explored are not exhaustive, and they only serve here as a demonstration. We highlight that finding the optimal settings poses an open question.

4 Conclusions

We introduced DHNs as generalisation of Hopfield networks. By providing an energy function and generalising results of Bruck (1990) we supported the claim that DHNs with classification function as activation are natural multidimensional analogues of classical Hopfield networks. We also provide equivalence between DHNs and graph clusterings methods like the Louvain method and the Newman's method. This enables us to generalise these two approaches, which gives rise to promising and effective modularity maximisers. Note that we did not discuss associative content-addressable memory capacity of DHNs even for classification function, or methods for training DHNs. We leave the exploration of this subject as a promising direction for further research.

Acknowledgments and Disclosure of Funding

The research presented in this paper was supported by the Polish National Centre for Research and Development (NCBiR) under Grant No. POIR.01.01.01-00-0695/19.

References

Allaire, G. and Kaber, S. M. (2008). *Numerical Linear Algebra*. Springer, New York, NY, 1st edition.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Bruck, J. (1990). On the convergence properties of the hopfield model. *Proceedings of the IEEE*, 78(10):1579–1585.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558.

Newman, M. E. J. (2006a). Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3).

Newman, M. E. J. (2006b). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582.

Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113.

Rychalska, B., Babel, P., Goluchowski, K., Michalowski, A., and Dabrowski, J. (2021). Cleora: A simple, strong and scalable graph embedding scheme. *CoRR*, abs/2102.02302.

A Appendices

A.1 Proofs of convergence theorems

In this appendix we are supplying the reader with rigorous proofs of Theorems 2.2 and 2.3.

Theorem 2.2. *Suppose that $W \in \mathbb{M}_{n\times n}(\mathbb{R})$ is a symmetric matrix with nonnegative entries on the diagonal and $B \in \mathbb{M}_{n\times d}(\mathbb{R})$. Then the function*

$$V\left(X\right) = -Tr\left(X^TWX + 2X^TB\right)$$

is an energy function function for the multidimensional Hopfield network (W, B, cl) operating in serial mode.

For the proof we need the following technical lemma:

Lemma A.0.1. *Let $W \in \mathbb{M}_{n\times n}(\mathbb{R})$ be a symmetric matrix and let $B \in \mathbb{M}_{n\times d}(\mathbb{R})$. Suppose that $X, \Delta \in \mathbb{M}_{n\times d}(\mathbb{R})$. Define*

$$H = WX + B$$

Let $V : \mathbb{M}_{n\times d}(\mathbb{R}) \to \mathbb{R}$ be the function given by

$$V\left(X\right) = -\mathrm{Tr}\left(X^TWX + 2X^TB\right)$$

Then

$$V(X+\Delta) - V(X) = -2\cdot\mathrm{Tr}\left(\Delta^TH\right) - \mathrm{Tr}\left(\Delta^TW\Delta\right)$$

Proof of the lemma. We have

$$V(X+\Delta) - V(X) =$$
$$= -\mathrm{Tr}\Big((X+\Delta)^TW(X+\Delta) + 2(X+\Delta)^TB\Big) + \mathrm{Tr}\Big(X^TWX + 2X^TB\Big) =$$
$$= -\mathrm{Tr}\Big(X^TW\Delta + \Delta^TWX + \Delta^TW\Delta + 2\Delta^TB\Big)$$

Since W is symmetric, and the trace of a square matrix equals the trace of its transpose, we can write

$$\mathrm{Tr}(X^TW\Delta) = \mathrm{Tr}\Big(\left(X^TW\Delta\right)^T\Big) = \mathrm{Tr}(\Delta^TW^TX) = \mathrm{Tr}(\Delta^TWX)$$

Since Tr is linear, we obtain

$$\mathrm{Tr}\Big(X^T W\Delta + \Delta^T W X + \Delta^T W\Delta + 2\Delta^T B\Big) = \mathrm{Tr}\Big(2\Delta^T W X + 2\Delta^T B\Big) + \mathrm{Tr}\left(\Delta^T W \Delta\right) =$$

$$= 2\cdot \mathrm{Tr}\Big(\Delta^T (WX + B)\Big) + \mathrm{Tr}\left(\Delta^T W\Delta\right) = 2\cdot \mathrm{Tr}\left(\Delta^T H\right) + \mathrm{Tr}\left(\Delta^T W\Delta\right)$$

This proves the lemma. □

Proof of the theorem. Assume that (W, B, cl) operates in serial mode. Suppose that $X \in \mathbb{M}_{n\times d}(\mathbb{R})$ is the matrix of neuron states of (W, B, cl) at some time $t \in \mathbb{N}$. Suppose that the states of the neurons of (W, B, cl) at time $t+1$ are given by the rows of $X + \Delta$, where $\Delta \in \mathbb{M}_{n\times d}(\mathbb{R})$. According to Lemma A.0.1 we have

$$V\left(X+\Delta\right) - V\left(X\right) = -2\cdot \mathrm{Tr}\left(\Delta^T H\right) - \mathrm{Tr}\left(\Delta^T W\Delta\right)$$

where $H = WX + B$. Since the network is operating in serial mode, there is a unique neuron say k which is updated at timestamp t. Let $j_1, j_2 \in \{1, ..., d\}$ be such that

$$j_1 = \underset{j}{\mathrm{argmax}}\, H_{kj} = \underset{j}{\mathrm{argmax}} \left\{\sum_{i=1}^{n} W_{ki}X_{ij} + B_{kj}\right\},\ X_{kj_2} = 1$$

If $j_1 = j_2$, then $\Delta = 0$. Hence we may assume that $j_1 \neq j_2$. Then

$$\Delta_{kj_1} = 1,\ \Delta_{kj_2} = -1$$

and these are the only nonzero entries of Δ. Taking this into consideration we have

$$-2\cdot \mathrm{Tr}\left(\Delta^T H\right) - \mathrm{Tr}\left(\Delta^T W\Delta\right) = -2\cdot\left(H_{kj_1} - H_{kj_2}\right) - W_{j_1j_1} - W_{j_2j_2}$$

Since W has nonnegative entries on the diagonal, we derive that

$$-2\cdot\left(H(X)_{kj_1} - H(X)_{kj_2}\right) - W_{j_1j_1} - W_{j_2j_2} \leq -2\cdot\left(H_{kj_1} - H_{kj_2}\right)$$

Using the facts that

$$j_1 = \underset{j}{\mathrm{argmax}}\, H_{kj},\ X_{kj_2} = 1,\ j_1 \neq j_2,$$

we obtain $H_{kj_1} - H_{kj_2} > 0$. Thus, in summary, we proved that

$$V\left(X+\Delta\right) - V\left(X\right) \leq 0$$

and the equality holds if and only if states of neurons does not change during update at timestamp t. □

Theorem 2.3. *Let (W, B, cl) be a d-dimensional Hopfield network with n neurons. Suppose that W is symmetric. Then the following assertions hold.*

(1) *If W has nonnegative entries on the diagonal and (W, B, cl) operates in a serial mode, then the corresponding dynamical system converges to a stable state for every initial state of the neurons.*

(2) *If (W, B, cl) is operating in parallel mode, then the corresponding dynamical system converges to a cycle of length at most 2 for every initial state of the neurons.*

Proof. The assertion **(1)** is an immediate consequence of the existence of an energy function - which was proved in Theorem 2.2.

The proof of **(2)** relies on the same idea as the proof of the corresponding statement in Bruck (1990). Suppose that (W, B, cl) runs in a parallel mode with sequence of neurons states $\{X(t)\}_{t\in\mathbb{N}}$. Consider a d-dimensional Hopfield network $(\hat{W}, \hat{B}, \mathrm{cl})$ with $2n$ neurons, where

$$\hat{W} = \begin{pmatrix} 0 & W \\ W & 0 \end{pmatrix} \quad \hat{B} = \begin{pmatrix} B \\ B \end{pmatrix}$$

The network $(\hat{W}, \hat{B}, \mathrm{cl})$ is bipartite, with partitions

$$\{1, 2, ..., n\},\ \{1+n, 2+n, ..., n+n\}$$

Further, it's connection satisfies the following:

- Neuron i in the first partition is connected to neuron $j+n$ in the second partition by directed edge of weight W_{ij}.

- Neuron $j+n$ in the second partition is connected to neuron i in the first partition by directed edge of weight W_{ji}, which is also equal to W_{ij}.
- Neuron i in the first partition has bias vector B_i.
- Neuron $j+n$ in the second partition has bias vector B_j.

We show that there is a DHN with connections and biases as above, which in a serial mode is equivalent to (W, B, cl). To do so, we first provide an initial state for the network, followed by a sequence of serial mode operations, and we prove that the state of (W, B, cl) can be deduced from the state of the extended network. First, we set the state of the network at $t = 0$. We set:

$$\hat{X}_{i+n}(0) = X_i(0)$$

for every $i \in \{1, ..., n\}$ and we set $\hat{X}_i(0)$ to be an arbitrary vector in $\mathbb{R}^d$ for every $i \in \{1, ..., n\}$. Then, we update the nodes cyclically according to the following sequence.

$$1, 2..., n, 1+n, 2+n, ..., n+n$$

Using the architecture of $(\hat{W}, \hat{B}, F)$, one can prove by mathematical induction that

$$\hat{X}_i\left((2t+1)\cdot n\right) = X_i(2t+1),\ \hat{X}_{i+n}\left(2t \cdot n\right) = X_i(2t)$$

for each $i \in \{1, ..., n\}$ and $t \in \mathbb{N}$. Since $\hat{W}$ is symmetric with zero diagonal, we use **(1)** to derive that $\{\hat{X}(t)\}_{t\in\mathbb{N}}$ is in a stable state for all sufficiently large times t. Let $\hat{X} \in \mathbb{M}_{2n\times d}(\mathbb{R})$ be this stable state. Then

$$X_i(2t) = \hat{X}_{i+n} = X_i(2t+2)$$

and

$$X_i(2t+1) = \hat{X}_i = X_i(2t+3)$$

for each $i \in \{1, ..., n\}$ and for all sufficiently large times t. Thus $\{X(t)\}_{t\in\mathbb{N}}$ converges to a cycle of length 2. □

A.2 Greedy graph clustering methods

The work of Bruck (1990) provides an excellent starting point to understand the connection between Hopfield networks and graph cuts encoded by their states – running the Hopfield network yields cuts with smaller cut-values over iterations. In this section we present a generalisation of this result to multidimensional Hopfield networks with classification function activation. In particular, we prove that DHNs optimise graph clusterings in an analogous way to how Hopfield networks optimise graph cuts.

Definition A.1. Let n, d be positive integers and let G be a graph with set of nodes $\{1, ..., n\}$. *A d-clustering of* G is a partitioning of $\{1, ..., n\}$ into d disjoint subsets. The subsets are referred to as *clusters*.

Next we define the measure of quality of graph clusterings.

Definition A.2. Let n, d be positive integers and let G be a weighted graph with set of nodes $\{1, ..., n\}$ nodes and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. Let $\{c_1, ..., c_d\}$ be a d-clustering of G. Then

$$C_G\left(c_1, ..., c_d\right) = \sum_{k\neq l}\sum_{i\in c_k}\sum_{j\in c_l} W_{ij}$$

is *the d-cut value of* $\{c_1, ..., c_d\}$ *with respect to* W.

In order to relate graph clusterings with multidimensional Hopfield networks with cl activation we introduce the following special class of matrices.

Definition A.3. Let n, d be positive integers. A matrix in $\mathbb{M}_{n\times d}(\mathbb{R})$ with rows in $\mathbb{L}_d$ is a *clustering matrix*. The set of all clustering matrices in $\mathbb{M}_{n\times d}(\mathbb{R})$ is denoted by $\mathbb{K}_{n\times d}(\mathbb{R})$.

Remark A.4. According to Remark 1.7 we have $\mathrm{ran}(\mathrm{cl}) = \mathbb{K}_{n\times d}(\mathbb{R})$.

As the name suggests, every clustering matrix matrix $\mathbb{K}_{n\times d}(\mathbb{R})$ encodes a clustering of $\{1, ..., n\}$. Indeed, if X is a clustering matrix with n rows and d columns, then

$$c_k = \left\{i \in \{1, ..., n\} \,\middle|\, X_{ik} = 1\right\}$$

for $k \in \{1, ..., d\}$ is a clustering

Remark A.5. Note that there are multiple clustering matrices in $\mathbb{K}_{n\times d}(\mathbb{R})$ associated with the same clustering of $\{1, ..., n\}$. In fact, if $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ and $Y \in \mathbb{M}_{d\times d}(\mathbb{R})$ is an arbitrary permutation matrix, then the matrix XY corresponds to permuting the labels of the clusters defined by X, thus X and XY encode the same clustering.

We can express the d-cut value of a clustering in terms of an associated clustering matrix.

Fact A.6. *Let G be a graph with set of nodes $\{1, ..., n\}$ and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. Let d be a positive integer and let $\{c_1, ..., c_d\}$ be a d-clustering of G. If $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ is a clustering matrix encoding $\{c_1, ..., c_d\}$, then*

$$C_G\left(c_1, ..., c_d\right) = \mathrm{Vol}(G) - \mathrm{Tr}\left(X^T W X\right)$$

where

$$\mathrm{Vol}(G) = \sum_{i,j=1}^{n} W_{ij}$$

Proof. Note that

$$\mathrm{Tr}\left(X^T W X\right) = \sum_{k=1}^{d} \sum_{i,j\in c_k} W_{ij}$$

and since

$$\sum_{k=1}^{d} \sum_{i,j\in c_k} W_{ij} = \left(\sum_{k=1}^{d} \sum_{i,j\in c_k} W_{ij} + \sum_{k\neq l} \sum_{i\in c_k} \sum_{j\in c_l} W_{ij}\right) - \sum_{k\neq l} \sum_{i\in c_k} \sum_{j\in V_l} W_{ij} =$$

$$= \underbrace{\sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij}}_{\mathrm{Vol}(G)} - \underbrace{\sum_{k\neq l} \sum_{i\in c_k} \sum_{j\in c_l} W_{ij}}_{C_G(c_1,...,c_d)}$$

we derive that

$$C_G\left(c_1, ..., c_d\right) = \mathrm{Vol}(G) - \mathrm{Tr}\left(X^T W X\right)$$

□

Using this observation and results from preceding sections, we present the following results.

Corollary A.7. *Let G be a graph with set of nodes $\{1, ..., n\}$ and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. Suppose that W has nonnegative entries on the diagonal. Then the d-dimensional Hopfield network $(W, 0, \mathrm{cl})$ with n neurons operating in a serial mode decreases the d-cut value for the d-clustering of G encoded by its matrix of neuron states at each step.*

Proof. The proof follows immediately from Fact A.6 and Theorem 2.2.

□

For the case when B is nonzero, we first need to introduce some notation. Consider a weighted graph with nodes set $\{1, ..., n + d\}$ and edge weight matrix in block form

$$\begin{pmatrix} W & B \\ B^T & U \end{pmatrix}$$

where $W \in \mathbb{M}_{n\times n}(\mathbb{R}), B \in \mathbb{M}_{n\times d}(\mathbb{R}), U \in \mathbb{M}_{d\times d}(\mathbb{R})$ and W, U are symmetric. We denote the graph obtained this way by $G\left(W, B, U\right)$. Next let $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ be a clustering matrix. Then the matrix written in the block form

$$\begin{pmatrix} X \\ I_d \end{pmatrix}$$

is called the canonical extension of X. Note that the canonical extension of X is a clustering matrix in $\mathbb{K}_{(n+d)\times d}(\mathbb{R})$. In particular, if $X \in \mathbb{K}_{n\times d}(\mathbb{R})$, then its canonical extension encodes a d-clustering of $G(W, B, U)$.

Theorem A.8. *Let (W, B, cl) be a d-dimensional Hopfield network with n neurons and let $U \in \mathbb{M}_{d\times d}(\mathbb{R})$ be a symmetric matrix. Suppose that W has nonnegative entries on the diagonal. Then the following assertions hold.*

(1) *If (W, B, cl) operates in a serial mode, then the d-clusterings of $G(W, B, U)$ encoded by the canonical extensions of successive matrices of neuron states of (W, B, cl) have decreasing d-cut values.*

(2) *A state X of (W, B, cl) is stable for every serial mode operation if and only if the d-cut value of the d-cut encoded by the canonical extension of X in $G(W, B, U)$ cannot be decreased by moving any of the first n nodes of $G(W, B, U)$ to another cluster.*

(3) *If*

$$U_{ij} = \begin{cases} -\kappa & \textit{if } i \neq j \\ 0 & \textit{otherwise} \end{cases}$$

for sufficiently large $\kappa \in \mathbb{R}$, then there exists a state X of (W, B, cl) which is stable for every serial mode operation, and the d-clustering of $G(W, B, U)$ encoded by the canonical extension of X has globally minimal d-cut value.

Proof. For brevity we denote $G(W, B, U)$ by G. Let V be the energy function of d-dimensional Hopfield network (W, B, cl) described in Theorem 2.2. Pick a matrix $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ and let $\{c_1, ..., c_d\}$ be a clustering of G encoded by the canonical extension of X. Then by Fact A.6 we have

$$C_G(c_1, ..., c_d) = \mathrm{Vol}\,(G) - \mathrm{Tr}\left(\begin{pmatrix} X^T & I_d^T \end{pmatrix} \cdot \begin{pmatrix} W & B \\ B^T & U \end{pmatrix} \cdot \begin{pmatrix} X \\ I_d \end{pmatrix}\right) =$$

$$= \mathrm{Vol}\,(G) - \mathrm{Tr}(U) + V(X)$$

From this observation we can immediately infer **(1)** and **(2)**.
In order to prove **(3)** consider $m, M \in \mathbb{R}$ such that

$$m \leq -\mathrm{Tr}\left(X^T W X + 2X^T B Y\right) \leq M$$

for all $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ and $Y \in \mathbb{K}_{d\times d}(\mathbb{R})$. Fix $\kappa \in \mathbb{R}$ satisfying

$$m + \kappa > M$$

and we assume that U is of the special form as indicated in **(3)**. Let $c_1, ..., c_d$ be a clustering of G with minimal d-cut value. Let $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ and $Y \in \mathbb{K}_{d\times d}(\mathbb{R})$ be matrices such that

$$\begin{pmatrix} X \\ Y \end{pmatrix} \in \mathbb{K}_{(n+d)\times d}(\mathbb{R})$$

encodes $\{c_1, ..., c_d\}$. According to Fact A.6 we have

$$C_G(c_1, ..., c_d) = \mathrm{Vol}\,(G) - \mathrm{Tr}\left(\begin{pmatrix} X^T & Y^T \end{pmatrix} \cdot \begin{pmatrix} W & B \\ B^T & U \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \end{pmatrix}\right) =$$

$$= \mathrm{Vol}\,(G) - \mathrm{Tr}\left(X^T W X + Y^T B^T X + X^T B Y + Y^T V Y\right) =$$

$$= \mathrm{Vol}\,(G) - \mathrm{Tr}\left(X^T W X\right) - 2\mathrm{Tr}\left(X^T B Y\right) - \mathrm{Tr}\left(Y^T U Y\right) =$$

$$= \mathrm{Vol}\,(G) - \mathrm{Tr}\left(X^T W X + 2X^T B Y\right) + \kappa \cdot \sum_{k=1}^{d} \sum_{i \neq j} Y_{ik} Y_{jk}$$

Now if $Y_{ik} = 1 = Y_{jk}$ for some $j \neq i$ and $k \in \{1, ..., d\}$, then

$$C_G(c_1, ..., c_d) \geq \mathrm{Vol}\,(G) + m + \kappa > \mathrm{Vol}\,(G) + M$$

This contradicts the fact that C_G achieves it's global minimum for $\{c_1, ..., c_d\}$. Hence Y is a permutation matrix and thus $Y^{-1} = Y^T$ is also a permutation matrix. Since

$$\begin{pmatrix} X \\ Y \end{pmatrix} \cdot Y^{-1} = \begin{pmatrix} XY^{-1} \\ I_d \end{pmatrix}$$

and according to Remark A.5, we derive that the canonical extension of XY^{-1} encodes $\{c_1, ..., c_d\}$. By **(2)** we infer that XY^{-1} is a stable state for (W, B, cl) for every serial mode of operation, and this completes the proof of **(3)**. □

A.3 Details on Louvain and Newman's methods

Intuitively, the modularity value associated with an edge measures the difference between the weight of an edge compared to a null-hypothesis obtained by a probabilistic approximation of the underlying graph Newman and Girvan (2004). The null hypothesis is usually obtained using the configuration model. Given a graph, each edge is split in half, then each half edge (often referred to as stub) is rewired randomly with any other stub in the network. Thus, the baseline weight for a pair of nodes is given by the expected number of edges between the pair of nodes in a random rewiring of the stubs. A major advantage of approximating a graph in this way, is that the degrees of the nodes, and hence the degree distribution of the graph is preserved.

If an edge has a higher weight than the one expected based on the null-hypothesis, it is assumed to indicate an important connection (and a positive modularity value is assigned), whereas if the weight is smaller than the expected, it indicates smaller than expected connection between the two nodes (and a negative modularity value is assigned). More exactly:

Definition A.9. Let n be a positive integer and let G be a graph with set of nodes $\{1, ..., n\}$ and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. Consider the matrix $Q \in \mathbb{M}_{n\times n}(\mathbb{R})$ given by

$$Q_{ij} = \frac{1}{\mathrm{Vol}(G)}\left(W_{ij} - \frac{k_i k_j}{\mathrm{Vol}(G)}\right)$$

where $\mathrm{Vol}(G) = \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}$ and $k_i = \sum_{j=1}^{n} W_{ij}$. Then Q is *a modularity matrix of* G.

This way, we can define the modularity of a clustering of G, as the sum of the intra-cluster modularities:

Definition A.10. Let n be a positive integer and let G be a graph with set of nodes $\{1, ..., n\}$ and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. The *modularity of a d-clustering* $\{c_1, ..., c_d\}$ *of* G is defined as

$$Q\left(c_1, ..., c_d\right) = \sum_{k=1}^{d}\sum_{i,j\in c_k} Q_{ij}$$

Remark A.11. With the notation as in the definition above we have

$$Q\left(c_1, ..., c_d\right) = \sum_{k=1}^{d}\sum_{i,j\in c_k} Q_{ij} = \sum_{i,j=1}^{n} Q_{ij} - \sum_{k\neq l}\sum_{i\in c_k}\sum_{j\in c_l} Q_{ij}$$

and hence maximizing modularity of G is the same as finding the minimal d-cut value of a graph which has the same set of vertices as G, but has edge weight matrix Q.

The Louvain method Blondel et al. (2008) is a popular graph clustering method, based around the heuristic idea of greedy local search, using the modularity of the clustering as an objective function to maximise. Strictly speaking, after reaching a local optimum, the Louvain method merges nodes in the same cluster together, but this part of the algorithm is irrelevant for our analysis, thus in the sequel we ignore it. In order to introduce the Louvain method we need some further notation. Let G be a graph with nodes $\{1, ..., n\}$ and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. Pick a node j of G. We denote

$$\mathcal{N}_j = \left\{i \in \{1, ..., n\} \setminus \{j\} \,\middle|\, W_{ij} > 0\right\}$$

Now suppose in addition that we have a clustering $\{c_i\}_{i=1}^{d}$ of G. Then we denote

$$\mathcal{NC}_j = \left\{m \in \{1, ..., d\} \,\middle|\, \mathcal{N}_j \cap c_m \neq \emptyset\right\}$$

Now the Louvain method is presented in Listing 2.

Listing 2: Louvain method in Blondel et al. (2008)

```
initialise the clustering $\{c_i\}_{i=1}^{d}$[1]
pick $u \in 1, \ldots, n$
    $m^* \leftarrow \mathrm{argmax}_{m\in\mathcal{NC}_u} Q(c_1 \setminus \{u\}, \ldots, c_{m-1} \setminus \{u\}, c_m \cup \{u\}, c_{m+1} \setminus \{u\}, \ldots, c_d \setminus \{u\})$
    move node $u$ to cluster $c_{m^*}$
```

Remark A.12. In Blondel et al. (2008) the initialisation takes the form $\{c_i\}_{i=1}^{n}$ with $c_i = \{i\}$.

In order to express the Louvain method as DHN running in serial mode, we first simplfy the operational logic of the algorithm.

Listing 3: Louvain method with simplified operational logic

```
initialise the clustering $\{c_i\}_{i=1}^{d}$
pick $u \in 1, \ldots, n$
    $m^* \leftarrow \mathrm{argmax}_{m} Q(c_1 \setminus \{u\}, \ldots, c_{m-1} \setminus \{u\}, c_m \cup \{u\}, c_{m+1} \setminus \{u\}, \ldots, c_d \setminus \{u\})$
    move node $u$ to cluster $c_{m^*}$
```

Now we are going to analyze Listing 3 further, and show that under some mild assumptions on the graph G, the algorithm is operationally equivalent (in the same state they produce the same output) to the original algorithm (i.e. Listing 2). We start by some simple observation.

Proposition A.13. *Let G be a graph with nodes $\{1, ..., n\}$ and edge weight matrix $W \in \mathbb{M}_{n\times n}(\mathbb{R})$. Suppose that $\{c_i\}_{i=1}^{d}$ is a clustering of G. Fix a node u of G. Then*

$$\underset{m}{\mathrm{argmax}}\, Q\left(c_1 \setminus \{u\}, \ldots, c_{m-1} \setminus \{u\}, c_m \cup \{u\}, c_{m+1} \setminus \{u\}, \ldots, c_d \setminus \{u\}\right) =$$

$$= \underset{m}{\mathrm{argmax}} \sum_{j\in c_m\setminus\{u\}} Q_{uj}$$

[1] In practice this initialisation can take many forms. See for example Remark A.12.

Proof. We have

$$Q\left(c_1 \setminus \{u\}, \ldots, c_{m-1} \setminus \{u\}, c_m \cup \{u\}, c_{m+1} \setminus \{u\}, \ldots, c_d \setminus \{u\}\right) =$$

$$= 2 \cdot \operatorname{Tr}(Q) + \sum_{k=1}^{d} \sum_{i,j \in c_k \setminus \{u\}, i \neq j} Q_{ij} + 2 \cdot \sum_{j \in c_m \setminus \{u\}} Q_{ju}$$

Since

$$2 \cdot \operatorname{Tr}(Q) + \sum_{k=1}^{d} \sum_{i,j \in c_k \setminus \{u\}, i \neq j} Q_{ij}$$

does not depend on m, the result follows. □

Next we show that Listing 2 and Listing 3 are operationally equivalent.

Theorem A.14. *Let G be a graph with nodes $\{1, ..., n\}$, edge weight matrix $W \in \mathbb{M}_{n \times n}(\mathbb{R})$ and modularity matrix Q. Assume that the following assertions hold.*

(1) *W has nonnegative entries.*

(2) *W has zeros on the diagonal (no self-loops in G).*

(3) *Each node in G has positive degree.*

Then one can replace third line in Listing 3 with

$$m^* = \operatorname*{argmax}_{m \in \mathcal{NC}_u} Q\left(c_1 \setminus \{u\}, \ldots, c_{m-1} \setminus \{u\}, c_m \cup \{u\}, c_{m+1} \setminus \{u\}, \ldots, c_d \setminus \{u\}\right)$$

without affecting the operation of the method.

Proof. In general (without any assumptions on weights of G) we have

$$\sum_{j=1}^{n} Q_{uj} = 0$$

Since $W_{uu} = 0$ and degree k_u of u is positive, we have $Q_{uu} < 0$. It follows that

$$\sum_{m=1}^{d} \sum_{j \in c_m \setminus \{u\}} Q_{uj} = \sum_{j \neq u} Q_{ju} = -Q_{uu} > 0$$

Hence if

$$m^* = \operatorname*{argmax}_{m} \sum_{j \in c_m \setminus \{u\}} Q_{uj}$$

then

$$\sum_{j \in c_{m^*} \setminus \{u\}} Q_{uj} > 0$$

and this is only possible if $Q_{uj} > 0$ for some $j \in c_{m^*} \setminus \{u\}$. Since $k_j, k_u, \operatorname{Vol}(G)$ are positive, this implies that $W_{uj} > 0$. Thus $j \in \mathcal{N}_u$ and we deduced that $m^* \in \mathcal{NC}_u$. Now it suffices to invoke Proposition A.13. □

Notice that, in this equivalent formulation of the Louvain method, steps $(2 - 4)$ in 3 closely resemble the update rule of a DHN with cl activation. More formally:

Proposition A.15. *Let G be a graph with nodes $\{1, ..., n\}$, edge weight matrix $W \in \mathbb{M}_{n \times n}(\mathbb{R})$ and modularity matrix Q. Suppose that $\tilde{Q}$ is the same matrix as Q but with zeros on the diagonal. Consider an n-dimensional Hopfield network $(\tilde{Q}, 0, \mathrm{cl})$ with n neurons and pick $u \in \{1, ..., n\}$. If X is a clustering matrix, then running a single serial update of $(\tilde{Q}, 0, \mathrm{cl})$ for neuron u and matrix of neuron states X produces the same result as updating clustering encoded by X by Listing 3 to node u.*

$$\begin{tikzcd}
c \arrow[r, "\text{Louvain update}"] & c' \\
X \arrow[u, dashed] \arrow[r, "\text{DHN update}"] & X' \arrow[u, dashed]
\end{tikzcd}$$

Figure 1: Figure illustrating the claim of Proposition A.15.

Proof. Let $X \in \mathbb{K}_{n\times d}(\mathbb{R})$ be a clustering matrix encoding clustering $\{c_1, ..., c_d\}$ of G and fix $u \in \{1, ..., d\}$. By Proposition A.13 we have

$$\operatorname*{argmax}_m Q\left(c_1 \setminus \{u\}, \ldots, c_{m-1} \setminus \{u\}, c_m \cup \{u\}, c_{m+1} \setminus \{u\}, \ldots, c_d \setminus \{u\}\right) =$$

$$= \operatorname*{argmax}_m \sum_{j \in c_m \setminus \{u\}} Q_{uj} = \operatorname*{argmax}_m \sum_{j \in c_m} \tilde{Q}_{uj} = \operatorname*{argmax}_m \sum_{j=1}^{n} \tilde{Q}_{uj} \cdot X_{jm}$$

Therefore, updating u according to the serial update rule for $(\tilde{Q}, 0, \mathrm{cl})$ and updating it by means of update in Listing 3 yields the same result.

□

Theorem A.14 and Proposition A.15 together form a rigorous restatement of Proposition 3.1. Here we note the following result.

Corollary A.16. *Using the same setting as in Theorem A.14, DHN $(\tilde{Q}, 0, \mathrm{cl})$ running in serial mode with I_n as the initial matrix of neuron states runs one whole iteration of the Louvain method with initialization as in Blondel et al. (2008).*

Proof. Since $I_n \in \mathbb{K}_{n\times n}(\mathbb{R})$ is a clustering matrix corresponding to the state when every vertex is in a different cluster, thus initial neuron states matrix of $(\tilde{Q}, 0, \mathrm{cl})$ encodes the initial state of the Louvain method by Remark A.12. According to Theorem A.14 and Proposition A.15 this property is preserved after every iteration. □

Another popular method frequently applied for finding clusters with high modularity scores is associated to Newman Newman (2006b,a). Let G be a graph as above and let Q be its modularity matrix. Newman's method relies on the fact, that finding a 2-clustering of G with maximal modularity can be written in terms of maximising quadratic form

$$s^T Q s$$

subject to constraint $s \in \{-1, 1\}^n$. Since Q is real and symmetric $n \times n$ matrix, it has a complete set of orthonormal eigenvectors, say $\mathbf{u}_1, ..., \mathbf{u}_n$, with corresponding eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n$. Then, for any vector $s \in \mathbb{R}^n$, we can write

$$s = a_1\mathbf{u}_1 + \cdots + a_n\mathbf{u}_n$$

where $a_i \in \mathbb{R}$. Then we have

$$s^T Q s = \left(\sum_{i=1}^{n} a_i \mathbf{u}_i\right)^T \cdot Q \cdot \left(\sum_{i=1}^{n} a_i \mathbf{u}_i\right) = \sum_{i=1}^{n} a_i^2 \lambda_i$$

If $\|s\|_2 = 1$ and since $\mathbf{u}_i$ are orthogonal, we derive that $\sum_{i=1}^{n} a_i^2 = 1$. It follows that

$$\mathbf{u}_1 = \operatorname*{argmax}_{\|s\|_2=1} s^T Q s$$

Thus one can approach solving

$$\max_{s \in \{-1,1\}^n} s^T Q s$$

by first finding the eigenvector of Q with the largest eigenvalue, and then finding the element of $\{-1, 1\}^n$ which, in geometric terms, aligns with it the most. That is:

$$\operatorname*{argmax}_{s \in \{-1,1\}^n} s^T \mathbf{u}_1$$

This has the closed form solution given by sgn $(\mathbf{u}_1)$ where sgn is applied component-wise (see Newman (2006b)). Under optimal circumstances the method is guaranteed to converge quickly. A major drawback of the algorithm, is that while spectral methods provide both a conceptually simple and easy-to-solve framework for finding a graph cut with high modularity, currently clustering into more than two parts is achieved by iteratively splitting clusters, as solving for more than two clusters is computationally hard Newman (2006a). In practice, calculating the leading eigenvector is achieved by applying the power-method (Allaire and Kaber, 2008, pp. 194-198) on Q. By construction, Q can be written as:

$$Q = W - \frac{kk^T}{\mathrm{Vol}(G)}$$

Where $k = (k_1, ..., k_n)^T$ is the degree vector of G (see Definition A.9). This way, assuming that W is sparse, which is very often the case in real-world applications, the time complexity of multiplying by Q reduces significantly. The listing below shows the pseudo-code implementation of the method.

Listing 4: Newman method

```
1   input $Q$
2   initialise $\mathbf{v}$ randomly
3   until $\mathbf{v}/\|\mathbf{v}\|_2$ converges
4       $\mathbf{v} = Q\mathbf{v}$
5   return $\operatorname{sgn}(\mathbf{v})$
```

Remark A.17. In implementations, the 4th line of Listing 4 is replaced by

$$\mathbf{v} = Q\mathbf{v}/||Q\mathbf{v}||_2$$

due to practical reasons. From the purely theoretical point of view, which is our main concern, this modification does not influence the output. Therefore, we decide to get rid of it.

Now, consider the 1-dimensional Hopfield network $(Q, \mathbf{0}, \mathrm{id})$ running in parallel mode. According to Remark 1.4, the network in parallel mode is equivalent to lines 3-4 in Listing 4. After convergence[2], the resulting vector, say $\mathbf{v}$, can't be used for clustering yet, since it's elements are continuous real values, therefore we apply sgn, to obtain the cut most resembling the output of $(Q, \mathbf{0}, \mathrm{id})$. Expressing the Newman method this way allows us to extend the method to the case of multi-label clustering, simply by considering a DHN of higher dimensions. When collapsing the final matrix of neuron states, though, this time our target space is the space of clustering matrices of appropriate dimensions. We provide following proposition about cl, that is a natural generalisation of the claim that $\max_{s\in\{-1,1\}^n} s^T\mathbf{u}_1$ has closed form solution $\operatorname{sgn}(\mathbf{u}_1)$.

Proposition A.18. *Let $M \in \mathbb{M}_{n\times d}(\mathbb{R})$. Suppose that $\mathrm{cl}(M)$ denotes the matrix obtained by application of* cl *to rows of M. Then*

$$\mathrm{cl}(M) = \underset{S\in\mathbb{K}_{n\times d}(\mathbb{R})}{\operatorname{argmax}}\ \mathit{Tr}\left(S^TM\right)$$

Proof. Clearly $\mathrm{cl}(M) \in \mathbb{K}_{n\times d}(\mathbb{R})$ by Remark A.4. Now, consider any $S \in \mathbb{K}_{n\times d}(\mathbb{R})$. Let $\sigma_S : \{1, \ldots, n\} \to \{1, \ldots d\}$ be a map given by formula

$$S_{ij} = \begin{cases} 1 & \text{if } \sigma_S(i) = j \\ 0 & \text{otherwise} \end{cases}$$

and note that

$$\mathrm{Tr}\left(S^TM\right) = \sum_{i=1}^{m}\sum_{j=1}^{n} S^T_{ij}M_{ji} = \sum_{i=1}^{m}\sum_{j=1}^{n} S_{ji}M_{ji} = \sum_{j=1}^{n} M_{j\sigma_S(j)}$$

By definition of cl we have

$$\sum_{j=1}^{n} M_{j\sigma_S(j)} \le \sum_{j=1}^{n} M_{j\sigma_{\mathrm{cl}(M)}(j)}$$

and hence

$$\mathrm{Tr}\left(S^TM\right) \le \mathrm{Tr}\left(\mathrm{cl}(M)^TM\right)$$

for every $S \in \mathbb{K}_{n\times d}(\mathbb{R})$. This completes the proof. □

Remark A.19. The proposition above can be also reformulated in the following way. Note that $\mathbb{M}_{n\times d}(\mathbb{R})$ is a Hilbert space with scalar product

$$\langle M_1, M_2\rangle = \mathrm{Tr}\left(M_1^TM_2\right)$$

which induces Frobenius norm

$$\|M\|_F = \sqrt{\mathrm{Tr}\left(M^TM\right)} = \sqrt{\sum_{i,j=1}^{n} M_{ij}^2}$$

for $M \in \mathbb{M}_{n\times d}(\mathbb{R})$. Then Proposition A.18 is equivalent to the fact that the map $\mathrm{cl} : \mathbb{M}_{n\times d}(\mathbb{R}) \twoheadrightarrow \mathbb{K}_{n\times d}(\mathbb{R})$ satisfies

$$\|\mathrm{cl}(M) - M\|_F \le \|S - M\|_F$$

for all $S \in \mathbb{K}_{n\times d}(\mathbb{R})$.

[2] Convergence is understood as the 'direction' of X being stuck in a cycle. More precisely, we halt the loop when $\|\hat{X}(t) - \hat{X}(t-k)\| < \varepsilon$, where t is the current timestamp, $k \in \{1, ..., N\}$ for some fixed $N \in \mathbb{N}$, $\varepsilon > 0$, $\hat{X} = X/\|X\|$ and $\|.\|$ is the Frobenius norm . The exact values the parameters ε and N can be finetuned based on the nature of the application.

Using this information it is straightforward to write Newman's method as a DHN. When extending to multiple dimensions, though, one has to take care that the extra dimensions are somehow 'coupled', otherwise they all converge to the leading eigenvector. See the main text for a possible solution.