IMPLICIT NEURAL REPRESENTATION OF GRAPHONS FOR ASYMPTOTIC EXTREMAL PROBLEMS IN GRAPH THEORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine learning is opening new directions in mathematical discovery, from automatic provers for IMO problems, to new insights and progress for mathematical conjectures guided by machine learning techniques. We extend this line of research by proposing a neural architecture that tackles asymptotic extremal problems in graph theory. For an optimisation question on graphs with n vertices, an asymptotic extremal problem investigates the limiting behaviour of the answer to the question as n tends to infinity. We start with a well-known fact that these problems are often equivalently formulated as optimisation problems over *graphons*, which are symmetric measurable functions from $[0,1]^2$ to $[0,1]$. We represent graphons as neural networks from $[0,1]^2$ to $[0,1]$, and solve graphon-optimisation problems by gradient descent. Because optimal graphons for these problems are usually discontinuous, naively using a standard neural architecture and a standard learning algorithm does not lead to mathematically-interesting near-optimal solutions. To overcome this challenge, we propose a new neural architecture inspired by diffusion models, which uses multiple varying-scale sinusoidal encodings of a given input. Also, to deal with a constraint in those graphon-optimisation problems effectively, our architecture includes a constraint solver that finds a solution of an empirical counterpart of such a constraint, and backpropages the gradient signal over the solver using implicit differentiation. For some generalised Turán problems and domination exponent problems that were solved by extensive human effort, our method rediscovers known optimal graphons without human intervention. For some open problems of the same kind, it finds novel candidate optima and counterexamples, which may be of interest to the mathematics community.

1 INTRODUCTION

Machine learning is opening new directions in mathematical discovery, from automatic provers for IMO problems (Polu et al., 2023; Trinh et al., 2024; Huang & Yang, 2025), and autoformalisation tools that translate human proofs into machine-checkable form (Wu et al., 2022; Jiang et al., 2023), to new insights and progress for mathematical conjectures guided by machine learning techniques (Wagner, 2021b; Davies et al., 2021; He et al., 2024). In this paper, we extend this line of research by proposing a neural architecture that tackles problems in extremal graph theory.

A central object in extremal graph theory is the homomorphism density $t(H, G)$ between two finite undirected graphs H and G. It measures the probability that a uniform random map from the vertex set $V(H)$ of H to the vertex set $V(G)$ of G becomes a homomorphism from H to G. For instance, for each n, the constrained optimisation problem

$$\max_{G} \; t(K_2, G) \;\text{ subject to }\; t(K_3, G) = 0 \;\text{ and }\; |V(G)| = n \tag{1}$$

is one of the most basic problems in extremal graph theory. Here K_m denotes the complete graph with m vertices, so K_2 and K_3 represent an edge and a triangle, respectively. The problem asks for finding a graph G that maximises the density of an edge (denoted as K_2) without having a triangle (denoted as K_3) as a subgraph. Although the answer to this problem is well-known and is, in fact, a 120-year-old theorem of Mantel (Mantel, 1907), a large number of similar optimisation problems

involving homomorphism density are still open, and are actively investigated in the extremal graph theory community (Sidorenko, 1993c; Borgs et al., 2006; Razborov, 2008).

One common approach for tackling such challenging optimisation problems with homomorphism density is to work on their asymptotic counterparts, where we consider an optimisation problem parameterised by the size n of finite graphs G, and analyse where the optimal value of the problem converges as n tends to infinity. It turns out that such asymptotic optimisation problems can be phrased in terms of continuous optimisation problems over graphons, which are just symmetric measurable functions from $[0, 1]^2$ to $[0, 1]$, and form limiting objects of increasing sequences of dense graphs (Lovász, 2012). By moving from discrete optimisation problems over finite graphs to continuous optimisation problems over graphons, one can compactify the search space, simplify the problems, and let them be studied with tools from continuous mathematics and optimisations[1].

Our goal is to solve these challenging optimisation problems over graphons from extremal graph theory using deep learning. We propose a particular architecture of neural networks that represents graphons in the style of implicit neural representation (Park et al., 2019; Mescheder et al., 2019; Chen & Zhang, 2019). That is, graphons are represented as neural networks from $[0, 1]^2$ to $[0, 1]$.

Our architecture is designed to handle two difficulties of the graphon optimisation problems. First, known optimal graphons are usually discontinuous and have the form of step functions. The unknown optimal graphons are expected to have similar properties. Second, the optimisation problems often have constraints. For instance, the optimisation problem over graphs G in Eq. (1) constrains the triangle homomorphism density of G to be 0, and its counterpart over graphons has a similar constraint involving the graphon version of the homomorphism density. To address the first difficulty, our architecture uses the multi-frequency multi-layer input encoding, and to tackle the second, it performs a form of constraint solving, while backpropagating the gradient signal over this constraint solving using implicit differentiation, as in deep equilibrium models (Bai et al., 2019). The architecture also comes with a particular training algorithm that achieves sample efficiency by exploiting the symmetries of the given graphon-optimisation problems.

We demonstrate the effectiveness of our architecture and its training algorithm experimentally with well-known constrained or unconstrained graphon optimisation problems. We show that our method recovers some famous nontrivial results in extremal graph theory, namely, the minimisation of the triangle density and also that of the 5-cycle density under the constraint on the edge density (Bollobas, 1975; Goodman, 1959; Lovász & Simonovits, 1983; Fisher, 1989; Razborov, 2008). We then use our method to obtain novel results, namely, the minimisation of the homomorphism density of the Petersen graph and that of the 7-cycle density under the same edge density constraint, and also a potential counterexample to the conjectured inequality between exponentiated homomorphism densities of the Heawood graph and the Desargues graph.

The rest of the paper is organised as follows. We start by recalling key concepts and asymptotic extremal problems in graph theory in §2. Then, we explain our neural-network architecture for those extremal problems and its training algorithm in §3. Finally, we describe our experimental evaluation of this architecture and the algorithm in §4. The evaluation shows that our method can recover some nontrivial known results using our neural networks and training algorithm, and also that it brings novel insights into certain unresolved asymptotic extremal problems in graph theory. The appendix includes discussion on related works, the missing proof, and the details of our experiments.

2 Background

For a positive integer n, we let $[n]$ be the set of integers $\{1, 2 \ldots, n\}$. In what follows, a graph means an undirected finite simple graph, i.e., an undirected graph on a finite vertex set that does not have an edge connecting a vertex to itself and has at most one edge between two vertices. For a graph G, we write $V(G)$ for its vertex set, and $E(G)$ for its edge set. Also, we let $v(G)$ and $e(G)$ be the cardinalities of $V(G)$ and $E(G)$, respectively. We use a few well-known graphs and the standard notations for them. For examples, K_n denotes the complete graph on n vertices (i.e., the graph with the vertex set $[n]$ and the edge set $\{\{i, j\} : i \neq j\}$), and C_n denotes the cycle of length n (i.e., the graph with the vertex set $[n]$ and the edge set $\{\{i, (i + 1)\} : 1 \leq i \leq n - 1\} \cup \{\{n, 1\}\}$).

[1]We point out that the simplification of the formulation itself rarely addresses the fundamental difficulties of those problems, despite offering a better language to resolve them.

Let H and G be graphs. A homomorphism from H to G is a map ϕ from $V(H)$ to $V(G)$ that preserves the edge relation, i.e., satisfies $\{\phi(u), \phi(v)\} \in E(G)$ for all $\{u, v\} \in E(H)$. The **homomorphism density** from H to G, a key concept in the extremal graph theory, represents the probability that a uniform random map from $V(H)$ to $V(G)$ becomes a homomorphism. It is denoted by $t(H, G)$, and has the following analytic form:

$$t(H,G) = \frac{1}{v(G)^{v(H)}} \sum_{\phi: V(H)\to V(G)} \prod_{\{u,v\}\in E(H)} \mathbb{I}[\{\phi(u), \phi(v)\} \in E(G)]. \tag{2}$$

A large body of results in extremal graph theory analyses the optimisation problems of the form:

$$\liminf_{n\to\infty} \left(\min_{G} \; f(t(H_1,G), ..., t(H_k,G)) \;\text{ s.t. }\; v(G) = n \wedge \bigwedge_{j=1}^{\ell} g_j(t(F_1,G), ..., t(F_m,G)) = 0 \right) \tag{3}$$

where $H_1, \ldots, H_k, F_1, \ldots, F_m$ are graphs and $f, g_1, \ldots, g_\ell$ are functions of reals. We focus on two important special cases of these optimisation problems. The first is to find the smallest homomorphism density of a given graph H subject to the fixed edge density p:

$$\liminf_{n\to\infty} \left(\min_{G} \; t(H,G) \;\text{ s.t. }\; v(G) = n \;\wedge\; t(K_2,G) = p \right). \tag{4}$$

This is a well-known generalisation of both Turán's theorem (Turán, 1941) and the Erdős–Stone–Simonovits theorem (Erdős & Stone, 1946; Erdos & Simonovits, 1966), both classical results in extremal graph theory. In parituclar, for any H and $p \leq 1 - (1/(\chi(H) - 1))$, where $\chi(H)$ is the chromatic number of H, the solution to (4) is zero. If $p > 1 - (1/(\chi(H) - 1))$, there are only a handful of examples of H for which the solution to (4) is known, e.g., $H = K_3$ by Razborov (2008), $H = K_4$ by Nikiforov (2011), and $H = K_r$ by Reiher (2016). Even for C_5, the solution is known only for $p = 1 - 1/k$, where k is a positive integer (Bennett et al., 2020).

The second type does not have any constraint and has the following form: for graphs H_1 and H_2,

$$\liminf_{n\to\infty} \left(\min_{G} \; t(H_1,G)^{m_2} - t(H_2,G)^{m_1} \;\text{ s.t. }\; v(G) = n \right). \tag{5}$$

Here m_1 and m_2 are positive integers depending on H_1 and H_2, such as $(e(H_1), e(H_2))$. Often, we are interested in whether the asymptotic optimum of Eq. (5) is non-negative or not. The celebrated conjecture of Sidorenko (1993a;b) predicts that the answer is yes whenever H_1 is bipartite, $H_2 = K_2$, and $(m_1, m_2) = (e(H_1), e(H_2))$. More systematic studies for general pairs of graphs (H_1, H_2) have also been carried out in recent work (Stoner, 2022; Conlon & Lee, 2024; Behague et al., 2025).

Our results in the paper rely on an important reformulation of the asymptotic optimisation problem in Eq. (3) using graphons. Mathematically, a **graphon** W is just a symmetric measurable function from $[0,1]^2$ to $[0,1]$. Intuitively, it represents the adjacency matrix of a graph that has an uncountable number of vertices in $[0,1]$ and has edges labelled with probabilities. We extend the homomorphism density to graphons W, and define, for a given graph H with the vertex set $[n]$,

$$t(H,W) = \int_{[0,1]^n} \prod_{\{i,j\}\in E(H)} W(x_i, x_j)\, dx_1 ... dx_n = \mathbb{E}_{x_1,\ldots,x_n \overset{\text{iid}}{\sim} \text{Unif}[0,1]} \left[\prod_{\{i,j\}\in E(H)} W(x_i, x_j) \right]. \tag{6}$$

Then, it is known that the solution of the problem in Eq. (3) is equal to

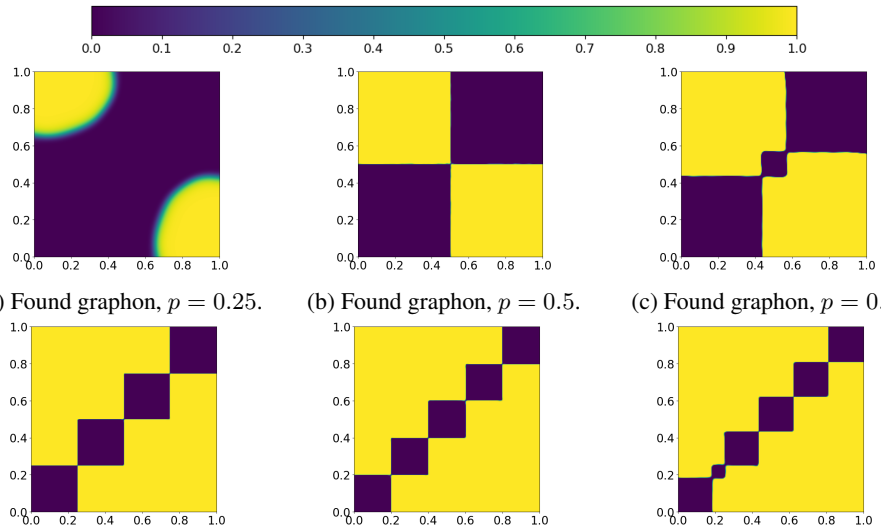
$$\min_{W} \; f(t(H_1,W), \ldots, t(H_k,G)) \;\text{ s.t. }\; \bigwedge_{j=1}^{\ell} g_j(t(F_1,W), \ldots, t(F_m,W)) = 0 \tag{7}$$

whenever $f, g_1, \ldots, g_\ell$ are continuous. We use this graphon formulation of the problem in the paper.

3 Extremal Problems with Implicit Neural Representation

We tackle the following graphon versions of the optimisation problems in Eqs. (4) and (5):

$$\min_{W} \; t(H,W) \;\text{ s.t. }\; t(K_2,W) = p, \qquad \text{and} \qquad \min_{W} \left(t(H_1,W)^{m_2} - t(H_2,W)^{m_1} \right), \tag{8}$$



(a) Found graphon, $p = 0.25$. (b) Found graphon, $p = 0.5$. (c) Found graphon, $p = 0.6$.

(d) Found graphon, $p = 0.75$. (e) Found graphon, $p = 0.8$. (f) Found graphon, $p = 0.82$.

Figure 1: Graphons found for K_3 density minimisation under an edge density constraint.

for graphs H, H_1, H_2, a real $p \in (0, 1)$, and positive integers m_1, m_2. Here W ranges over graphons.

The high-level idea of our method is to represent W as a neural network f_θ, rephrase the above optimisation problems to those over the network parameters θ, and solve the rephrased problems using a gradient-based training algorithm. A naive implementation of this idea does not work for multiple reasons. First, known optimal graphons are usually step functions, the type of functions that are difficult to learn using standard neural architectures, such as MLPs, because such architectures often have a bias towards smooth functions. Second, the four occurrences of $t(_, W)$ in the problems are defined in terms of intractable integrals, but efficiently estimating them or their exponentiations with samples requires problem-specific insight. Finally, the first problem has a constraint, and the usual way of encoding this constraint as a regulariser is suboptimal. A better alternative is to ensure that a constraint or its sample-based counterpart holds by the construction of the architecture. In this section, we present our architecture for the implicit neural representation of graphons, and its training algorithm based on gradient descent. In so doing, we explain how our architecture and training algorithm overcome the three challenges described above.

Architecture Our architecture for neural networks representing graphons is a variant of MLPs that has residual connection and, more importantly, injects the layer-specific sinusoidal encoding of the input at every layer. When the architecture is used for solving the problem of the first type in Eq. (8) or other similarly-constrained graphon optimisation problems, it includes a constraint solver. Concretely, our architecture assumes that we are given the number of layers $L \geq 2$, the number of hidden units in each layer d, a differentiable activation function $\phi : \mathbb{R} \to \mathbb{R}$, a standard deviation $\sigma \in (0, \infty)$ to initialise network parameters, and an increasing function $s : [L] \to (0, \infty)$ that maps each layer to its input-encoding scale; in our applications, $s(\ell) = \ell$ or $s(\ell) = 2^{\ell-1}$. Then, the architecture defines a graphon-representing network $f_\theta[c]$ with a special parameter $c \in \mathbb{R}$ and the other parameters θ as follows. For all $2 \leq \ell \leq L$,

$$f_\theta[c] : [0,1]^2 \to [0,1], \qquad g_\theta^{(1)}, g_\theta^{(\ell)} : \{\mathbf{x} \in [0,1]^2 : \mathbf{x} = (x_1, x_2) \text{ for some } x_1 \leq x_2\} \to \mathbb{R}^d$$

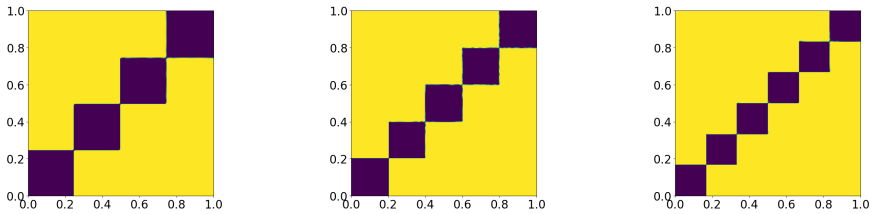
$$\begin{aligned} f_\theta[c](x_1, x_2) &= \operatorname{sigmoid}\big(V^{(L)} g_\theta^{(L)}(\min(x_1, x_2), \max(x_1, x_2)) + c\big), \\ g_\theta^{(1)}(\mathbf{x}) &= \phi\big(W^{(1)} \sin(U^{(1)}\mathbf{x}) + b^{(1)}\big), \\ g_\theta^{(\ell)}(\mathbf{x}) &= g_\theta^{(\ell-1)}(\mathbf{x}) + \phi\big(V^{(\ell-1)} g_\theta^{(\ell-1)}(\mathbf{x}) + W^{(\ell)} \sin(U^{(\ell)}\mathbf{x}) + b^{(\ell)}\big). \end{aligned}$$

The network parameters

$$c \in \mathbb{R}, \qquad b^{(\ell)} \in \mathbb{R}^d, \qquad V^{(L)} \in \mathbb{R}^{1 \times d}, \qquad V^{(\ell')}, W^{(\ell)} \in \mathbb{R}^{d \times d}, \qquad \text{and} \qquad U^{(\ell)} \in \mathbb{R}^{d \times 2}$$

for $\ell' \in [L-1]$ and $\ell \in [L]$ are initialised by

$$c,\ b_i^{(\ell)},\ V_{1,i}^{(L)},\ V_{i,j}^{(\ell')},\ W_{i,j}^{(\ell)} \overset{\text{iid}}{\sim} \operatorname{Unif}[-1/\sqrt{2d}, 1/\sqrt{2d}], \qquad U_{i,k}^{(\ell)} \overset{\text{iid}}{\sim} \operatorname{Unif}[s(\ell)/\sqrt{2}, s(\ell)/\sqrt{2}],$$



(a) Found graphon, K_4 case. (b) Found graph, K_5 case. (c) Found graph, K_6 case.

Figure 2: Graphons found for K_n density minimisation under edge density constraint with $p{=}1-\frac{1}{n}$.

for all $i, j \in [d]$ and $k \in [2]$. The θ represents all these parameters except c. Note that by the use of $\min$ and $\max$, we have $f_\theta[c](x_1, x_2) = f_\theta[c](x_2, x_1)$ for all x_1, x_2, and so $f_\theta[c]$ denotes a graphon.

The architecture has two notable features. The first is the repeated injection of the sinusoidal encoding of the input (x_1, x_2) into each network layer, where the initialisation variance of the embedding increases with the layer depth. This feature is inspired by the diffusion model's trick for effectively modelling a distribution with multiple modes (Salmona et al., 2022). The diffusion model generates a multi-modal distribution in several steps, with early steps focusing on the low-frequency components of the distribution and the late steps on the high-frequency components (Dieleman, 2024). We conjecture that multiple modes in a distribution and multiple steps in a step function pose a similar modelling challenge, and adopt the diffusion model's strategy in our architecture. We want the early layers of our network $f_\theta[c]$ to model the low-frequency parts of a target graphon, and the late layers to model the high-frequency parts. To achieve this desired behaviour, we increase the intialisation variance of the sinusoidal encoding of the inputs with the layer depth, which has the effect of increasing the frequency of the embedding with the layer depth as well (Tancik et al., 2020).

The second feature is the use of a constraint solver to set the network parameter c. When the network $f_\theta[c]$ is used to solve the problem of the first type in Eq. (8) so that it has to satisfy the edge density constraint $t(K_2, f_\theta[c]) = p$ for some $p \in (0, 1)$, the parameter c is not learnt by our training algorithm, but is determined by a constraint solver. More precisely, in that case, we assume that at each training step or the inference time of the network, a collection of K samples of pairs

$$\mathcal{E} = \{(x_1^{(k)}, x_2^{(k)})\}_{k\in[K]} \in [0,1]^2 \ \text{ with } \ (x_1^{(k)}, x_2^{(k)}) \sim \mathrm{Unif}([0,1]\times[0,1]) \ \text{ for all } k \in [K]$$

is given, before $f_\theta[c]$ is applied to any input pairs. Then, the network runs a constraint solver to find an appropriate value of the parameter c:

$$\textsc{Solve}_\theta(\mathcal{E}, p) = c_0 \ \text{ such that } \left(\frac{1}{|\mathcal{E}|}\sum_{(x_1,x_2)\in\mathcal{E}} f_\theta[c_0](x_1, x_2)\right) = p. \tag{9}$$

The right-hand side of the above equation is an unbiased estimate of $t(K_2, f_\theta[c_0])$. So, the equation is an empirical counterpart of the original constraint $t(K_2, f_\theta[c_0]) = p$. The equation has a unique solution since its right-hand side as a function of c_0 is continuous and strictly increasing, and converges to 0 and 1 as c_0 tends to $-\infty$ and ∞, respectively. We use a variant of Newtonian method to solve the constraint; see Algorithm 1 in the appendix for the description of the variant. After finding the solution c^* in the constraint-solving step, the network $f_\theta[c^*]$ is applied to other input pairs.

We point out that when the network $f_\theta[c]$ is applied to the problem of the second type in Eq. (8), the parameter c is learnt by our gradient-based training algorithm, just like the other parameters θ.

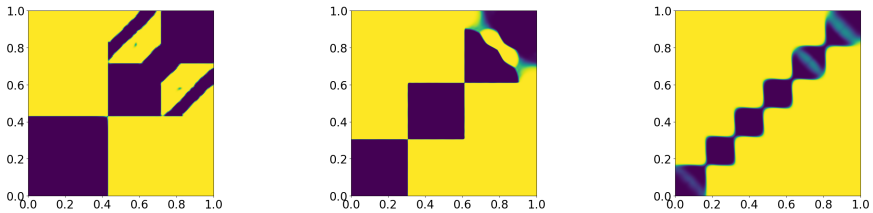
Estimation of Homomorphism Density Recall the homomorphism density from a graph H with vertex set $[n]$ to a graphon $f_\theta[c]$ is: $t(H, f_\theta[c]) = \mathbb{E}_{x_1,\ldots,x_n \overset{\text{iid}}{\sim} \mathrm{Unif}[0,1]}[\prod_{\{i,j\}\in E(H)} f_\theta[c](x_i, x_j)]$. The core component of our training algorithm is our estimator of this homomorphism density.

Our estimator for $t(H, f_\theta[c])$ is parameterised by a collection $\mathcal{S}$ of permutations on the vertex set $[n]$ of H. Given this parameter, it samples $N \times n$ independent samples from the uniform distribution on $[0, 1]$, and computes the estimate $\widehat{t}(H, f_\theta[c])$ as follows:

$$x_i^{(k)} \overset{\text{iid}}{\sim} \mathrm{Unif}[0,1] \text{ for } (k,i) \in [N]\times[n], \qquad \widehat{t}(H, f_\theta[c]) = \frac{1}{N|\mathcal{S}|}\sum_{\substack{k\in[N]\\ \pi\in\mathcal{S}}} \prod_{\substack{\{\pi(i),\pi(j)\}\\ \in E(H)}} f_\theta[c]\Big(x_{\pi(i)}^{(k)}, x_{\pi(j)}^{(k)}\Big). \tag{10}$$

Table 1: Homomorphism densities of C_5 in graphons, which are found by our method and discretisation or by other method. The edge density constraint is specified by $p \in (0, 1)$.

	$p = 0.6$	$p = 0.7$	$p = 11/13$
Conjectured	0.0666156	0.1631493	0.4333230
Ours	**0.0666704 (+0.08%)**	**0.1631983 (+0.03%)**	0.4334453 (+0.03%)
Hand-crafted	0.07075 (+6.21%)	0.1653186 (+1.33%)	**0.4334225 (+0.03%)**



(a) Found graphon, $p = 0.6$. (b) Found graphon, $p = 0.7$. (c) Found graphon, $p = 11/13$.

Figure 3: Graphons found for C_5 density minimisation under an edge density constraint.

The most interesting part of the estimator is $\mathcal{S}$. When $\mathcal{S} = \{\mathrm{id}\}$, the estimator is just the standard Monte-Carlo estimator of the target expectation with N independently-sampled sets $\{\{x_1^{(k)}, \ldots, x_n^{(k)}\}\}_{k \in [N]}$. But when $\mathcal{S}$ has at least two elements and $\mathcal{S}$ is chosen well, the estimator can, in the best case, achieve the level of performance with $N \times |\mathcal{S}|$ independently-sampled sets, while sampling only N sets. Also, two permutations π and π' in $\mathcal{S}$ may share some edges, i.e., $\{\pi(i), \pi(j)\} = \{\pi'(i'), \pi'(j')\} \in E(H)$, and such shared edges allow us to reduce the number of the network evaluation from $|\mathcal{S}| \cdot N \cdot e(H)$ to $C \cdot N \cdot e(H)$ for some $C < |\mathcal{S}|$. We use this additional power of our estimator when we experiment with large H's, such as Petersen graph.

The estimator can be extended to compute an unbiased estimate of any polynomial expression of $t(H_1, f_\theta[c]), \ldots, t(H_m, f_\theta[c])$, such as $t(H_1, f_\theta[c])^{m_2} - t(H_2, f_\theta[c])^{m_1}$. This is because we can eliminate the product of two homomorphism densities using the below formula:

$$t(H, f_\theta[c]) \times t(H', f_\theta[c]) = t(H \oplus H', f_\theta[c]) \tag{11}$$

where $H \oplus H'$ denotes the disjoint union of H and H'. Given a polynomial, we expand it to the linear combination of monomials, and eliminate the products of homomorphism densities using Eq. (11). The result is a linear combination of homomorphism densities. We estimate each term in this linear combination using our estimator and combine the estimated values linearly. We apply this extension of our estimator when we tackle the second problem in Eq. (8).

Training Algorithm Our training algorithm for the network $f_\theta[c]$ approximately solves a given graphon-optimisation problem using the estimated gradient of the optimisation objective.

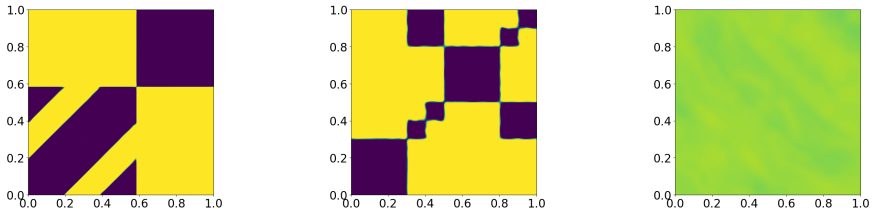
Consider the first problem in Eq. (8). Assume that we are given a graph H with vertex set $[n]$, and a real $p \in (0, 1)$. The goal in this case is to find a network $f_\theta[c]$ such that $t(H, f_\theta[c])$ is as small as possible while $t(K_2, f_\theta[c])$ is close to p. Our training algorithm achieves this goal by repeatedly setting c to an approximate solution c_θ of constraint $t(K_2, f_\theta[c]) = p$ and updating θ with an estimated gradient of $t(H, f_\theta[c])$. The details of this parameter-update step are given below:

$$x_i^{(k)} \overset{\text{iid}}{\sim} \mathrm{Unif}[0, 1] \;\text{ for }\; k \in [N] \;\text{ and }\; i \in [n],$$
$$\mathcal{D} = \{\{x_1^{(k)}, \ldots, x_n^{(k)}\} : k \in [N]\} \;\text{ and }\; \mathcal{E} = \{(x_i^{(k)}, x_j^{(k)}) : k \in [N] \wedge i, j \in [n] \wedge i < j\},$$
$$\theta_{t+1} \leftarrow \theta_t - \eta_t \cdot \nabla_\theta \, \widehat{t}_{\mathcal{D}}(H, f_\theta[c_\theta])\big|_{\theta = \theta_t} \;\text{ where }\; c_\theta = \textsc{Solve}_\theta(\mathcal{E}, p).$$

Here $\textsc{Solve}_\theta(\mathcal{E}, p)$ is a constraint solver from Eq. (9), and we add the subscript $\mathcal{D}$ in $\widehat{t}_{\mathcal{D}}$ to clarify the estimator's dependence on $\mathcal{D}$. We set the learning rate η_t by the cosine-annealing learning-rate scheduler (Loshchilov & Hutter, 2017) since the scheduler allows the training algorithm to alternate between searching for a structure and fine-tuning the boundary of the found structure. Note that datasets $\mathcal{D}$ and $\mathcal{E}$ are sampled fresh in each parameter-update step, instead of being sampled in the first step and reused subsequently. We do not follow this latter option since it can lead to overfitting of the network to the reused datasets. Note also that $\mathcal{E}$ reuses the sampled $x_i^{(k)}$'s in $\mathcal{D}$. This reuse lets us reduce the number of network evaluations $f_\theta[c_\theta]$ during gradient computation.

Table 2: Homomorphism densities of C_7 in graphons, which were found by our method and discretisation or by other baseline. The edge density constraint is specified by $p \in (0, 1)$.

	$p = 0.6$	$p = 0.7$	$p = 11/13$
Conjectured	0.0262613	0.0819155	0.31054965
Ours	**0.0262873 (+0.10%)**	**0.081972 (+0.07%)**	0.310558 (+0.0027%)
Hand-crafted	0.027556 (+4.93%)	0.082241 (+0.40%)	**0.310553 (+0.0012%)**



(a) Found graphon, $p = 0.6$. (b) Found graphon, $p = 0.7$. (c) Found graphon, $p = 11/13$.

Figure 4: Graphons found for C_7 density minimisation under an edge density constraint.

By the definition of our homomorphism density estimator, we have

$$\nabla_\theta \, \widehat{t}_{\mathcal{D}}(H, f_\theta[c_\theta]) = \frac{1}{N|\mathcal{S}|} \sum_{\substack{k \in [N], \pi \in \mathcal{S} \\ e = \{\pi(i), \pi(j)\} \in E(H)}} \left(\nabla_\theta \, f_\theta[c_\theta](x^{(k)}_{\pi(i)}, x^{(k)}_{\pi(j)}) \right) \prod_{\substack{\{\pi(u), \pi(v)\} \\ \in E(H) \setminus \{e\}}} f_\theta[c_\theta](x^{(k)}_{\pi(u)}, x^{(k)}_{\pi(v)}).$$

The gradient in the parameter-update step is computed using the above formula and the next lemma about $\nabla_\theta f_\theta[c_\theta]$, which is derived using implicit differentiation.

Lemma 1. *Let* $h_\theta(x_1, x_2) = V^{(L)} g^{(L)}_\theta(\min(x_1, x_2), \max(x_1, x_2))$. *Then, for all* $\mathbf{x} \in [0, 1]^2$,

$$\nabla_\theta \, f_\theta[c_\theta](\mathbf{x}) = \frac{\partial f_\theta[c](\mathbf{x})}{\partial \theta}\Big|_{c = c_\theta} - \mathrm{sigmoid}'(h_\theta(\mathbf{x}) + c_\theta) \sum_{\mathbf{y} \in \mathcal{E}} \frac{\mathrm{sigmoid}'(h_\theta(\mathbf{y}) + c_\theta)}{\sum_{\mathbf{z} \in \mathcal{E}} \mathrm{sigmoid}'(h_\theta(\mathbf{z}) + c_\theta)} \nabla_\theta \, h_\theta(\mathbf{y}),$$

where sigmoid$'$ *is the derivative of* sigmoid.

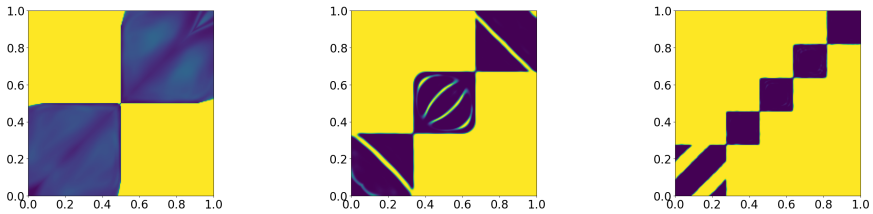
Now we move on to the second problem in Eq. (8). Let H_1 and H_2 be graphs with vertex sets $[n_1]$ and $[n_2]$, respectively. Consider positive integers m_1, m_2. The goal this time is to find a network $f_\theta[c]$ that minimises $t(H_1, f_\theta[c])^{m_2} - t(H_2, f_\theta[c])^{m_1}$. By Eq. (11), the optimisation objective can be rephrased to $t(H_1^{\oplus m_2}, f_\theta[c]) - t(H_2^{\oplus m_1}, f_\theta[c])$, where $H_1^{\oplus m_2}$ means the disjoint combination of the m_2 copies of H_1 (i.e., $H_1 \oplus \ldots \oplus H_1$ with H_1 appearing m_2 times) and $H_2^{\oplus m_1}$ is defined similarly. Our training algorithm performs optimisation by updating both θ and c of the network using gradient descent. The details of each parameter-update step are described below:

$$x^{(k)}_i, y^{(k)}_j \overset{\text{iid}}{\sim} \mathrm{Unif}[0, 1] \;\text{ for }\; k \in [N], \; i \in [m_2 \times n_1], \;\text{ and }\; j \in [m_1 \times n_2],$$
$$\mathcal{D} = \{\{x^{(k)}_1, \ldots, x^{(k)}_{m_2 \times n_1}\} : k \in [N]\} \;\text{ and }\; \mathcal{D}' = \{\{y^{(k)}_1, \ldots, y^{(k)}_{m_1 \times n_2}\} : k \in [N]\},$$
$$(\theta_{t+1}, c_{t+1}) \leftarrow (\theta_t, c_t) - \eta_t \cdot \nabla_{(\theta, c)} \left(\widehat{t}_{\mathcal{D}}(H_1^{\oplus m_2}, f_\theta[c]) - \widehat{t}_{\mathcal{D}'}(H_2^{\oplus m_1}, f_\theta[c]) \right)\Big|_{(\theta, c) = (\theta_t, c_t)}.$$

We sample datasets $\mathcal{D}$ and $\mathcal{D}'$ fresh at each step, instead of sampling them in the first step and reusing them subsequently. This is again to prevent the trained network from overfitting particular datasets.

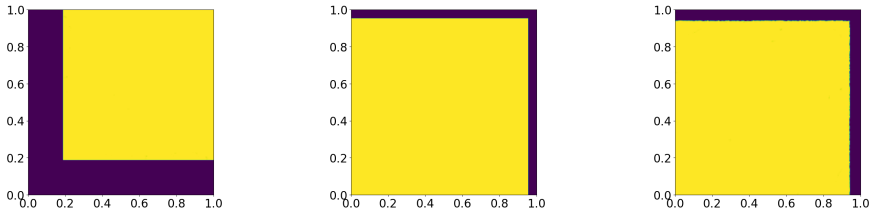
4 Experimental Results

We experimentally tested the effectiveness of our method with both resolved and unresolved problems in extremal graph theory, which have one of the forms in Eq. (8). The purposes of our experiments were to check whether the method can rediscover existing solutions of problems that are obtained by extensive human efforts, and also to see whether our method can find mathematically-interesting examples for unresolved problems. In this section, we explain the results of these experiments. We also describe our ablation study, which analyses the impact of components of our



(a) Found graphon, $p = 0.6$. (b) Found graphon, $p = 0.7$. (c) Found graphon, $p = 0.82$.

Figure 5: Graphons found for minimisation of the Petersen density under an edge density constraint.



(a) Graphon, $(r, s) = (3, 4)$. (b) Graphon, $(r, s) = (3, 5)$. (c) Graphon, $(r, s) = (4, 5)$.

Figure 6: Graphons found for the task of minimising $(t(K_r, W)^{\binom{s}{2}} - t(K_s, W)^{\binom{r}{2}})$ over W

network architecture and training algorithm on the performance of our method. Throughout the section, we visualise graphons as colour maps using the blue-to-yellow scheme in Fig. 1, where blue represents 0 and yellow 1. Also, among the graphons found by our method, we present those with the minimum optimisation-objective values. See Appendix D for the details of our experiments.

Minimisation of Homomorphism Density Under Edge Density Constraint We applied our method to the problem of minimising the homomorphism density of K_n under an edge density constraint $t(K_2, W) = p$. This is precisely the problem of the first type in Eq. (8) with K_n serving the role of the graph H there. When $n = 3$ and so K_n is a triangle, this optimisation problem has been tackled several times in the literature (Bollobas, 1975; Goodman, 1959; Lovász & Simonovits, 1983; Fisher, 1989; Razborov, 2008), and its full solution is obtained by Razborov (2008). For general K_n, the full answer to the problem was given only recently by Reiher (2016).

We first consider the case $n = 3$. When $p = 1 - 1/k$ for some integer $k \geq 2$, the optimal graphon is the complete k-partite graphon or any of its measure-preserving permutations. For $i \in [k]$, let $X_i = [(i-1)/k, i/k]$. The complete k-partite graphon W_k is defined as follows: $W_k(x_1, x_2) = 1$ if $x_1 \in X_i$ and $x_2 \in X_j$; otherwise, $W_k(x_1, x_2) = 0$. For a measure-preserving ϕ on $[0, 1]$, its permutation is $W_{k,\phi}(x_1, x_2) = W_k(\phi(x_1), \phi(x_2))$. Every optimum is of this form. When $1-1/k < p < 1-1/(k+1)$, the minimization admits multiple types of optima. One example is a variant W_p of the $(k+1)$-partite graphon, where the first k parts are equal in length but the last part X_{k+1} is shorter. Again, its permutations $W_{\phi,p}$ by measure-preserving ϕ on $[0, 1]$ are also optimal, though in this regime W_p and $W_{\phi,p}$ do not exhaust all optima.

We trained our network on six cases $p \in \{0.25, 0.5, 0.6, 0.75, 0.8, 0.82\}$. For $p \in \{0.25, 0.5, 0.75, 0.8\}$, our method produced near-optimal graphons (Figs. 1a, 1b, 1d and 1e) whose estimated objectives (via discretisation) are within $\pm 10^{-3}$ of the theoretical optima. When $p \in \{0.75, 0.8\}$, i.e. $p = 1 - 1/k$, the learned graphons are very close to the complete k-partite graphons W_k described above. In all these cases, permutations via measure-preserving maps yield equivalent objectives but more complex discontinuities, and our method consistently preferred the simpler representatives shown. For $p \in \{0.6, 0.82\}$, which satisfy $1-1/k < p < 1-1/(k+1)$ for respective k, the learned graphons (Figs. 1c and 1f) exhibit the core structure of the optimal W_p described above.

For K_n with $n > 3$, we applied our method to the following optimisation problems: $\min_W t(K_n, W)$ s.t. $t(K_2, W) = 1 - \frac{1}{n}$, for $n \in \{4, 5, 6\}$. The complete n-partite graphons and their permutations achieve optimal values of these problems. Despite the increasing complexity of these problems in the structures of their optimal graphons and the sample and computational complexities of our estimator, our method successfully found near-optimal graphons in Fig. 2, which show the desired complete n-partite structure.

Table 3: Estimated homomorphism densities of Heawood and Desargues (D), and their powered difference. The conjecture says the difference between exact densities is always positive.

	First candidate	Second candidate
$t(\text{Heawood}, W)$	$6.89875482111 \times 10^{-25}$	$8.6560174602121 \times 10^{-27}$
$t(D, W)$	$4.713366541252 \times 10^{-35}$	$5.856903258274 \times 10^{-38}$
$t(\text{Heawood}, W)^{\frac{10}{7}} - t(D, W)$	$-1.6657651864583 \times 10^{-35}$	$-9.5380064278 \times 10^{-42}$
$t(\text{Heawood}, W) - t(D, W)^{\frac{7}{10}}$	$-2.4624522111071 \times 10^{-25}$	$-9.86883402 \times 10^{-31}$

(a) First candidate counterexample. (b) Second candidate counterexample.

Figure 7: Graphons found by our method for the minimisation task induced by the conjectured Heawood vs Desargues inequality. These graphons are potential counterexamples.

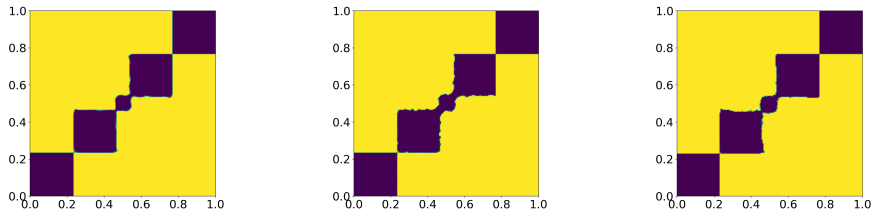
Unresolved Problems: Minimum Densities of C_5, C_7, and Petersen Graph We applied our method to three unresolved optimisation problems over graphons, which seek the minimisers of the C_5, C_7, and Petersen graph densities, respectively, that have a given edge density p.

For the C_5 density minimisation problem, partial results are known. When $p = 1 - 1/k$, Bennett et al. (2020) characterised the minimisers mathematically; for other p, they conjectured minimisers and optimum values instead. We applied our method to $p \in \{0.6, 0.7, 11/13\}$, which lie in this unresolved case. The learned graphons are shown in Fig. 3. To evaluate their C_5 densities, we discretised each to $2^{14} \times 2^{14}$ and computed the densities exactly (up to floating-point error). As shown in Table 1, the results closely match the conjectured optima with relative errors $< 0.1\%$, supporting the conjecture. Compared with our best-effort hand-optimised baselines, our method performs better for $p \in \{0.6, 0.7\}$ and comparably for $p = 11/13$. Notably, for $p = 0.6$ the learned graphon exhibits the conjectured structure, whereas for $p \in \{0.7, 11/13\}$ it does not, suggesting the existence of alternative minimisers.

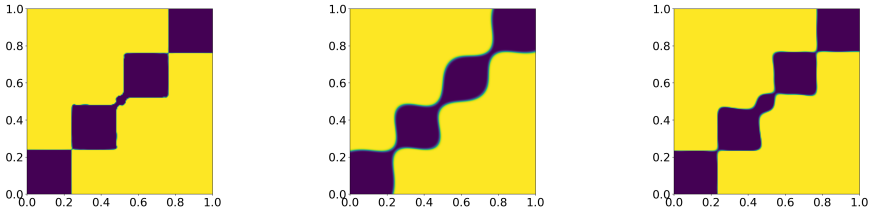
For the homomorphism density of C_7, no results are known or conjectured to the best of our knowledge. By generalising the conjecture of Bennett et al. (2020) on the C_5 density, we devised a new conjecture on the structure of a C_7-minimising graphon with a given edge density p and the optimum value achieved by this graphon. The details of our conjecture are given in Appendix F. Once again the C_7 densities of the graphons found by our method and discretisation have the C_7 densities that match the values predicted by our generalised conjecture with relative errors less than 0.2%. Also, our method outperformed the baseline for $p \in \{0.6, 0.7\}$. See Table 2. However, for $p = 11/13$, only nearly constant graphons that do not have any interesting structures were found by our method, which shows the limitation of our method.

For the miminisation of the homomorphism density of the Petersen graph, there are no known results or conjectures to the best of our knowledge, likely due to the size of the graph and the lack of a simple structure in it. We applied our method to this minimisation problem for $p \in \{0.6, 0.7, 0.82\}$. The graphons found by our method appear in Fig. 5. Due to the large size of the graph, we could not numerically discretise these graphons and compute their Petersen graph densities exactly (up to floating point errors). We instead estimated their densities with 2^{28} samples with a 95% bootstrap confidence interval, and compared the estimated values with those obtained by the baseline. The results are given in Table 8 in the appendix, which shows our method outperformed the baseline.

Normalised Inequalities between Homomorphism Densities We now describe our experiments with the problem of the second type in Eq. (8). The purpose of these experiments was to disprove, for some instances (H_1, H_2, m_1, m_2) of the problem, that the minimum values of the instances are non-negative, equivalently, the inequality $t(H_1, W)^{m_2} \geq t(H_1, W)^{m_1}$ between normalised homomorphism densities holds for all graphons W.



(a) Graphon found by our original method. (b) Graphon found by the first ResNet variant. (c) Graphon found by the second constant-s variant.



(d) Graphon found by the third variant of using a regulariser. (e) Graphon found by the forth variant with a low learning rate. (f) Graphon found by the fourth variant with a high learning rate.

Figure 8: Graphons found by our method and its variants on the triangle density minimisation task.

Our first task was to disprove the inequalities $t(K_r, W)^{\binom{s}{2}} \geq t(K_s, W)^{\binom{r}{2}}$ for $r, s \in \{3, 4, 5\}$ with $r < s$. We noted that such an inequality is equivalent to $(\min_W t(K_r, W)^{\binom{s}{2}} - t(K_s, W)^{\binom{r}{2}}) \geq 0$, and applied our method to find a graphon witnessing that the minimum value here is negative. For all cases of r and s, our method found graphons that strongly suggest the failure of the inequalities; the optimisation objectives at these graphons were estimated to be negative. See Table 9 in the appendix for the details. The found graphons appear in Fig. 6.

Our next task aims at disproving the open conjecture of Perarnau & Perkins (2018): for any graphon W and graph G consisting only of degree 3 vertices with shortest cycle length ≥ 6, we should have $t(\text{Heawood}, W)^{1/14} \geq t(G, W)^{1/v(G)}$. The Heawood graph is defined as in, e.g., Perarnau & Perkins (2018). We considered the Desargues graph D (García-Marco & Knauer, 2024), whose vertices all have degree 3 and whose shortest cycle has length 10, as a potential counterexample for G. Using our method, we identified two candidates graphons W (Table 3 and Fig. 7) that may violate the inequality.[2] Because D is large, we avoided discretisation and instead estimated homomorphism densities with 2^{28} samples.

Ablation Study We carried out the ablation study of our method with the triangle density minimisation task under an edge density constraint. In this task, we constrained the edge density to $p = 7/9$, which does not fall into the $1 - 1/k$ case and has multiple types of optimal graphons. We considered four variants of our method: (1) changing the architecture to ResNet with the sinusoidal encoding in the first layer; (2) setting the function s for per-layer input-encoding scales to a constant function; (3) enforcing the edge density constraint with a regulariser rather than using a constraint solver; (4) using a constant learning rate η instead of setting it according to the cosine-annealing scheduler. For fair comparison, we did a hyperparameter search for an appropriate constant for the function s in (2), the strength of the regulariser in (3), and an appropriate value of η in (4).

The graphons found by all these four variants are given in Fig. 8. The ResNet variant (1) and the constant-scale variant (2) fail to recover straight boundaries due to their biases towards smooth functions. For the variant with a regulariser (3), the found graphon fails to match the correct size of the smallest block. Similarly, in the absence of the cosine-annealing learning-rate scheduling, the found graphon fails to learn the five-block structure (when the learning rate is low) or fails to have straight boundaries (when the learning rate is high). These highlight the importance of alternating high and low learning rates in the cosine-annealing learning-rate scheduling. The results of evaluating the found graphons after discretisation appear in Table 10 in the appendix.

[2]We list both, but the first is stronger since the differences in the third and fourth rows of Table 3 scale comparably to the corresponding densities in the second and first rows.

REPRODUCIBILITY

The details on the experiments, for instance, the exact implementation of our algorithms or the set of hyperparameters sweeped are included in Appendix D. The Jupyter notebook to replicate our experiments is provided in the following anonymised repository `https://anonymous.4open.science/r/INR4AsymptoticExtremalGraphTheory-C390/`. The repository contains the code to reproduce our experiments for the triangle, 5-cycle, Petersen graph, the inequality between K_3 and K_4, and the inequality between the Desargues graph and the Heawood graph, together with the evaluation code that allows one to numerically compute or estimate the homomorphism densities of these graphs in a given graphon. We also provide the YAML files that specify the Anaconda environment to run the code. The rest of the experiments can be reproduced by changing a few lines in the provided code.

REFERENCES

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/file/01386bd6d8e091c2ab4c7c7de644d37b-Paper.pdf`.

Natalie Behague, Gabriel Crudele, Jonathan A. Noel, and Lina M. Simbaqueba. Sidorenko-type inequalities for pairs of trees. *Random Structures & Algorithms*, 67(1):e70026, 2025. doi: https://doi.org/10.1002/rsa.70026. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/rsa.70026`.

Patrick Bennett, Andrzej Dudek, Bernard Lidický, and Oleg Pikhurko. Minimizing the number of 5-cycles in graphs with given edge-density. *Combinatorics, Probability and Computing*, 29(1): 44–67, 2020. doi: 10.1017/S0963548319000257.

Bela Bollobas. Relations between sets of complete subgraphs. *Proceedings of the Fifth British Combinatorial Conference*, 01 1975.

Christian Borgs, Jennifer Chayes, László Lovász, Vera T Sós, and Katalin Vesztergombi. Counting graph homomorphisms. In *Topics in Discrete Mathematics: Dedicated to Jarik Nešetřil on the Occasion of his 60th Birthday*, pp. 315–371. Springer, 2006.

François Charton, Jordan S Ellenberg, Adam Zsolt Wagner, and Geordie Williamson. Patternboost: Constructions in mathematics with a little help from ai. *arXiv preprint arXiv:2411.00566*, 2024.

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

David Conlon and Joonkyung Lee. Domination inequalities and dominating graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 177(1):167–184, 2024. doi: 10.1017/S0305004124000185.

Alex Davies, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli. Advancing mathematics by guiding human intuition with ai. *Nature*, 600(7887):70–74, Dec 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-04086-x. URL `https://doi.org/10.1038/s41586-021-04086-x`.

Sander Dieleman. Diffusion is spectral autoregression, 2024. URL `https://sander.ai/2024/09/02/spectral-autoregression.html`.

P. Erdős and A. H. Stone. On the structure of linear graphs. *Bulletin of the American Mathematical Society*, 52:1087–1091, 1946.

Paul Erdos and Miklós Simonovits. A limit theorem in graph theory. *Studia Sci. Math. Hungar*, 1 (51-57):51, 1966.

Victor Falgas-Ravry and Emil R. Vaughan. Applications of the semi-definite method to the turán density problem for 3-graphs. *Combinatorics, Probability and Computing*, 22(1):21–54, 2013. doi: 10.1017/S0963548312000508.

David C. Fisher. Lower bounds on the number of triangles in a graph. *Journal of Graph Theory*, 13(4):505–512, 1989. doi: https://doi.org/10.1002/jgt.3190130411. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jgt.3190130411.

Ignacio García-Marco and Kolja Knauer. Beyond symmetry in generalized petersen graphs. *Journal of Algebraic Combinatorics*, 59(2):331–357, 2024.

A. W. Goodman. On sets of acquaintances and strangers at any party. *The American Mathematical Monthly*, 66(9):778–783, 1959. ISSN 00029890, 19300972. URL http://www.jstor.org/stable/2310464.

Yang-Hui He, Kyu-Hwan Lee, Thomas Oliver, and Alexey Pozdnyakov. Murmurations of elliptic curves. *Experimental Mathematics*, pp. 1–13, 2024.

Yichen Huang and Lin F Yang. Gemini 2.5 pro capable of winning gold at imo 2025. *arXiv preprint arXiv:2507.15855*, 2025.

Albert Qiaochu Jiang, Sean Welleck, Jin Peng Zhou, Timothee Lacroix, Jiacheng Liu, Wenda Li, Mateja Jamnik, Guillaume Lample, and Yuhuai Wu. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SMa9EAovKMC.

Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

L. Lovász and M. Simonovits. *On the number of complete subgraphs of a graph II*, pp. 459–495. Birkhäuser Basel, Basel, 1983. ISBN 978-3-0348-5438-2. doi: 10.1007/978-3-0348-5438-2_41. URL https://doi.org/10.1007/978-3-0348-5438-2_41.

László Miklós Lovász. Large networks and graph limits. In *Colloquium Publications*, 2012. URL https://api.semanticscholar.org/CorpusID:27057044.

Willem Mantel. Vraagstuk xxviii. *Wiskundige Opgaven met de Oplossingen*, 10(2):60–1, 1907.

Abbas Mehrabian, Ankit Anand, Hyunjik Kim, Nicolas Sonnerat, Matej Balog, Gheorghe Comanici, Tudor Berariu, Andrew Lee, Anian Ruoss, Anna Bulanova, et al. Finding increasingly large extremal graphs with alphazero and tabu search. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 6985–6993, 2024.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Konrad Mundinger, Max Zimmer, Aldo Kiem, Christoph Spiegel, and Sebastian Pokutta. Neural discovery in mathematics: Do machines dream of colored planes? In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=7Tp9zjP9At.

V. Nikiforov. The number of cliques in graphs of given order and size. *Transactions of the American Mathematical Society*, 363(3):1599–1618, 2011. ISSN 00029947. URL http://www.jstor.org/stable/41062613.

Olaf Parczyk, Sebastian Pokutta, Christoph Spiegel, and Tibor Szabó. New ramsey multiplicity bounds and search heuristics. *Foundations of Computational Mathematics*, pp. 1–38, 2024.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Guillem Perarnau and Will Perkins. Counting independent sets in cubic graphs of given girth. *Journal of Combinatorial Theory, Series B*, 133:211–242, 2018. ISSN 0095-8956. doi: https://doi.org/10.1016/j.jctb.2018.04.009. URL https://www.sciencedirect.com/science/article/pii/S0095895618300388.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=-P7G-8dmSh4.

Alexander A. Razborov. Flag algebras. *Journal of Symbolic Logic*, 72(4):1239–1282, 2007. doi: 10.2178/jsl/1203350785.

Alexander A. Razborov. On the minimal density of triangles in graphs. *Combinatorics, Probability and Computing*, 17(4):603–618, 2008. doi: 10.1017/S0963548308009085.

Alexander A. Razborov. *Flag Algebras: An Interim Report*, pp. 207–232. Springer New York, New York, NY, 2013.

Christian Reiher. The clique density theorem. *Annals of Mathematics*, 184(3):683–707, 2016. ISSN 0003486X. URL http://www.jstor.org/stable/44072027.

Antoine Salmona, Valentin De Bortoli, Julie Delon, and Agnes Desolneux. Can push-forward generative models fit multimodal distributions? In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 10766–10779. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/45f0d179ef7e10eb7366550cd4e574ae-Paper-Conference.pdf.

Alexander Sidorenko. Inequalities for functionals generated by bipartite graphs. *Discrete Math. Appl.*, 2:489–504, 1993a. ISSN 0924-9265. doi: 10.1515/dma.1992.2.5.489.

Alexander Sidorenko. A correlation inequality for bipartite graphs. *Graphs Combin.*, 9(2-4):201–204, 1993b. ISSN 0911-0119. doi: 10.1007/BF02988307.

Alexander Sidorenko. A correlation inequality for bipartite graphs. *Graphs and Combinatorics*, 9 (2):201–204, 1993c.

Cynthia Stoner. The graph density domination exponent. Preprint available at arXiv:2211.09870 [math.CO], 2022.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7537–7547. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf.

Trieu Trinh, Yuhuai Tony Wu, Quoc Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625:476–482, 2024. URL https://www.nature.com/articles/s41586-023-06747-5.

P. Turán. Eine extremalaufgabe aus der graphentheorie. *Matematikai és Fizikai Lapok*, 48:436–452, 1941.

Adam Zsolt Wagner. Constructions in combinatorics via neural networks, 2021a. URL https://arxiv.org/abs/2104.14516.

Adam Zsolt Wagner. Constructions in combinatorics via neural networks. *arXiv preprint arXiv:2104.14516*, 2021b.

Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. *Advances in neural information processing systems*, 35:32353–32368, 2022.

A Related Works

Razborov's *flag algebra* (Razborov, 2007) reformulates graph homomorphism inequalities in terms of 'sum of squares', i.e., the Cauchy–Schwarz inequality, over the algebra of graphs, where each finite graph is interpreted as a functional on the space of graphons. This opened up a new way to automate the applications of the Cauchy–Schwarz inequality by using semidefinite programming (SDP) solvers, which have been implemented by the package Flagmatic (Falgas-Ravry & Vaughan, 2013). While Flagmatic and flag algebra settled multiple open problems in extremal combinatorics for more than a decade (for a survey, see Razborov (2013)), the computational cost limits their application[3]. In particular, this limits its application to some instances of the problems we tried, including those problems involving the Petersen, Heawood, or Desargues graph.

For specific values of n without taking limits in Eqs. (4) and (5) or their generalisations, one may try to solve the discrete optimisation problem directly. A classical example is the local (Tabu) search with the help of reinforcement learning algorithms that still proves its usefulness in such problems, e.g., Parczyk et al. (2024); Mehrabian et al. (2024). As a direct application of reinforcement learning, Mehrabian et al. (2024) used AlphaZero to find the maximum number of edges in an n-vertex graph with no copies of C_3 and C_4. Recently, the so-called *deep cross entropy method* built upon reinforcement learning was introduced by Wagner (2021a). The method iteratively trains the network to learn a policy that promotes the generation of high quality examples, which succeeded in disproving two conjectures regarding the eigenvalue of the adjacency matrix and matching number or proximity. When applied to discrete problems, the deep cross entropy method may be more versatile than ours, as it allow optimisation of non-differentiable objectives like matching number. On the other hand, it only works on a predetermined graph size, whereas our method applies to asymptotic problems with the order of graphs tending to infinity. Recently, Patternboost by Charton et al. (2024) took a 'hybrid' approach that makes use of both local search and reinforcement learning. In particular, their illustrative example is to generate lower bound examples for Mantel's theorem, solving Eq. (4) without limit for $H = K_3$ up to $p \leq 1/2$ for small fixed values of n.

In terms of the method using implicit neural representation, the closest work to ours may be Mundinger et al. (2025) which applies implicit neural representations to the famous Hadwiger—Nelson problem: colouring all the points of $\mathbb{R}^2$ using the minimum number of colors, so that no two points of unit distance have the same color. Analogously to our method, they represent the colouring function with a neural network and recover a 6-colouring by minimising monochromatic edge density. With human post-processing, they construct two new 6-colourings that expand the known range of realisable distances from [0.415, 0.447] to [0.354, 0.657].

B Use of Large Language Models in Paper Writing

We used deep research or similar LLM-based search to find relevant papers and information about extremal graph theory. We discussed what we found with experts in extremal graph theory in order to ensure the correctness of the findings. While writing the paper, we used LLM-based autocomplete tools to fix grammatical errors. Finally, when we implement code for our experiments, we used LLM-based code generation tools in some parts of our implementation. For the LLM-generated parts, we checked their correctness by code reviewing and unit tests that we created.

C Unbiasedness of Our Density Estimator and Proof of Lemma 1

Lemma 2. *The estimator in Eq.* (10) *is unbiased, i.e.,* $\mathbb{E}[\widehat{t}(H, f_\theta[c])] = t(H, f_\theta[c])$.

[3]For instance, to derive nontrivial inequality on graph G, Flagmatic requires one to enumerate all graphs H with $v(H) = v(G) + 1$ and solve semidefinite programming over real vectors indexed by such graphs, but graphs H on more than eight vertices is currently unsupported due to its computational complexity.

Proof. We calculate the claimed equality as follows:

$$\begin{aligned}
\mathbb{E}\left[\widehat{t}(H, f_\theta[c])\right] &= \mathbb{E}\left[\frac{1}{N|\mathcal{S}|}\sum_{k\in[N],\pi\in\mathcal{S}}\left(\prod_{\{\pi(i),\pi(j)\}\in E(H)} f_\theta[c]\left(x^{(k)}_{\pi(i)}, x^{(k)}_{\pi(j)}\right)\right)\right] \\
&= \frac{1}{N|\mathcal{S}|}\sum_{k\in[N],\pi\in\mathcal{S}}\mathbb{E}\left[\prod_{\{\pi(i),\pi(j)\}\in E(H)} f_\theta[c]\left(x^{(k)}_{\pi(i)}, x^{(k)}_{\pi(j)}\right)\right] \\
&= \frac{1}{N|\mathcal{S}|}\sum_{k\in[N],\pi\in\mathcal{S}}\mathbb{E}_{x_1,\ldots,x_n\overset{\text{iid}}{\sim}\text{Unif}[0,1]}\left[\prod_{\{i,j\}\in E(H)} f_\theta[c](x_i, x_j)\right] \\
&= \frac{1}{N|\mathcal{S}|}\sum_{k\in[N],\pi\in\mathcal{S}} t(H, f_\theta[c]) = t(H, f_\theta[c]).
\end{aligned}$$

□

Proof of Lemma 1. Let $\mathbf{x} \in [0,1]^2$. Note that

$$\begin{aligned}
\nabla_\theta\ f_\theta[c_\theta](\mathbf{x}) &= \frac{\partial f_\theta[c](\mathbf{x})}{\partial\theta}\Big|_{c=c_\theta} + \frac{\partial f_\theta[c](\mathbf{x})}{\partial c}\Big|_{c=c_\theta}\nabla_\theta c_\theta \\
&= \frac{\partial f_\theta[c](\mathbf{x})}{\partial\theta}\Big|_{c=c_\theta} + \text{sigmoid}'(h_\theta(\mathbf{x}) + c_\theta)\nabla_\theta c_\theta
\end{aligned}$$

Thus, it suffices to show that

$$\nabla_\theta c_\theta = -\sum_{\mathbf{y}\in\mathcal{E}}\frac{\text{sigmoid}'(h_\theta(\mathbf{y}) + c_\theta)}{\sum_{\mathbf{z}\in\mathcal{E}}\text{sigmoid}'(h_\theta(\mathbf{z}) + c_\theta)}\nabla_\theta\, h_\theta(\mathbf{y}).$$

We prove the above equality by applying implicit differentiation to the following equation about c_θ, which holds by the choice of c_θ:

$$\frac{1}{|\mathcal{E}|}\sum_{\mathbf{y}\in\mathcal{E}} f_\theta[c_\theta](\mathbf{y}) = p.$$

First, we multiply both sides of the equation by $|\mathcal{E}|$, and differentiate them with respect to θ:

$$\left(\sum_{\mathbf{y}\in\mathcal{E}}\frac{\partial f_\theta[c](\mathbf{y})}{\partial\theta}\Big|_{c=c_\theta}\right) + \left(\sum_{\mathbf{z}\in\mathcal{E}}\frac{\partial f_\theta[c](\mathbf{z})}{\partial c}\Big|_{c=c_\theta}\nabla_\theta c_\theta\right) = 0$$

Thus,

$$\begin{aligned}
\nabla_\theta c_\theta &= -\left(\sum_{\mathbf{y}\in\mathcal{E}}\frac{\partial f_\theta[c](\mathbf{y})}{\partial\theta}\Big|_{c=c_\theta}\right)\Bigg/\left(\sum_{\mathbf{z}\in\mathcal{E}}\frac{\partial f_\theta[c](\mathbf{z})}{\partial c}\Big|_{c=c_\theta}\right) \\
&= -\sum_{\mathbf{y}\in\mathcal{E}}\frac{\text{sigmoid}'(h_\theta(\mathbf{y}) + c_\theta)}{\sum_{\mathbf{z}\in\mathcal{E}}\text{sigmoid}'(h_\theta(\mathbf{z}) + c_\theta)}\nabla_\theta\, h_\theta(\mathbf{y}),
\end{aligned}$$

as desired. □

D Details of Experiments

Every experiment was done on a single NVIDIA RTX A5000 GPU. The execution time depends on the size of the graph, from 30 minutes for a triangle to 10 hours for the Desargues graph.

For every experiment, we used a network with a width $d = 256$ and a depth $L = 6$, which was sufficient to express sharp boundaries and structural properties. The activation σ was either $\sin$ or GeLU nonlinearity, while there was no significant superiority between these two.

The set $\mathcal{S}$ used in Equation Eq. (10) was chosen depending on the graph; for instance, we used $S = \{id\}$ for complete graphs, as any permutation gives the same summand. For cycles, we used the representatives from the equivalence class $\Sigma_{V(C_k)}/\mathrm{Aut}(C_k)$, where $\mathrm{Aut}(C_k)$ is the automorphism group of C_k, giving $|\mathcal{S}| = (k-1)!/2$. For the other graphs, we used the set $\mathcal{S} = \{\sigma^k : k \in \mathbb{N}\}$ generated by some permutation $\sigma \in \Sigma(V(G))$, where σ is chosen so that the set

$$\{\{\sigma^k(v), \sigma^k(w)\} : \{v, w\} \in E(G) \text{ and } k \in \mathbb{N}\}$$

equals to the every pairs of vertices $\{\{v, w\} : v, w \in V(G) \text{ and } v \neq w\}$.

For the size of samples used in training, we used the maximal number of samples that fit in the GPU memory, ranging from 2^{12} to 2^{16}. For large graphs like the the Petersen graph, we repeated multiple forward passes to accumulate the gradients, so that the effective sample size is 2^{16}.

For the optimisation, we used Adam optimiser with $\beta_1 = 0.9, \beta_2 = 0.999$, which are the default values. We repeated our experiment with different learning rates $\eta \in \{10^{-3}, 5 \times 10^{-4}, 10^{-4}, 5 \times 10^{-5}\}$, and let the minimum learning rate in the learning rate scheduling to be 1% of the initial learning rate. We trained for 50,000 epochs, which is five cycles of cosine learning rate scheduling.

For the constraint solver, we found that the direct application of Newton's method without any modification is unstable, due to the vanishing gradient in most cases. Specifically, as most of the optimal graphons have the value either 0 or 1, the gradient of the sigmoid $\mathrm{sigmoid}(x) \cdot (1 - \mathrm{sigmoid}(x))$ is close to zero always. To resolve this, we use the clipped Newton's method, and to determine the clipping range, we find the upper and lower bounds of the solution c^*, and use their difference as the clipping range. We gradually decrease the clipping range as the iteration proceeds, which stabilises the training. The full algorithm is described in Algorithm 1.

Algorithm 1 Edge Density Constraint Solver

Require: $p \in [0, 1], y \in \mathbb{R}^N$
Ensure: $\frac{1}{N}\sum_{i=1}^{N} \mathrm{sigmoid}(y_i + c^*) \approxeq p$
  $c_1 \leftarrow \mathrm{sigmoid}^{-1}(p) - \frac{1}{N}\sum_{i=1}^{N} y_i$
  $D = \max_i\{y_i\} - \min_i\{y_i\}$
  while Not converged **do**
    $f(c_t) \leftarrow \frac{1}{N}\sum_{i=1}^{N} \mathrm{sigmoid}(y_i + c_t) - p$
    if $|f(c_t)| < \epsilon$ **then break**
    end if
    $s'_i \leftarrow \mathrm{sigmoid}(y_i + c_t)(1 - \mathrm{sigmoid}(y_i + c_t))$
    $f'(c_t) = \frac{1}{N}\sum_{i=1}^{N} s'_i$
    $c_{t+1} \leftarrow c_t - \mathrm{Clip}_{[-D/t, D/t]}\left(\frac{f(c_t)}{f'(c_t)}\right)$
  end while
  Return c_T

E Additional Tables

In this section, we provide the tables that summarise the numerical results of our experiments.

Table 4: The minimal K_3 density under the edge density constraint, and the numerically computed density of learnt graphon.

	Optimal value	Numerically Computed Density
$p = 0.25$	0	2.772029×10^{-13}
$p = 0.5$	0	2.166021×10^{-4}
$p = 0.6$	0.1415009	0.1415705
$p = 0.75$	0.375	0.3752523
$p = 0.8$	0.48	0.4802523
$p = 0.82$	0.52664016	0.5268536

Table 5: The minimal K_n density under edge density constraint $p = 1 - 1/n$, and the numerically computed density of learnt graphon.

	Optimal value	Numerically Computed Density	Estimated Density
$n = 4$	0.09375	0.0941397	0.09437804
$n = 5$	0.0384	0.04198146	0.03885484
$n = 6$	0.015432098765	0.02088445	0.01573410

In Tables 6–8, we add the other baseline, which parameterises a graphon with a fixed discretisation and optimises this discretised weighted graph with gradient descent and Adam optimiser.

Table 6: Homomorphism densities of C_5 in graphons, which are found by our method and discretisation or by other baselines. The edge density constraint is specified by $p \in (0, 1)$.

	$p = 0.6$	$p = 0.7$	$p = 11/13$
Conjectured	0.0666156	0.1631493	0.4333230
Ours	**0.0666704 (+0.08%)**	**0.1631983 (+0.03%)**	0.4334453 (+0.03%)
Hand-crafted	0.07075 (+6.21%)	0.1653186 (+1.33%)	**0.4334225 (+0.03%)**
GD	0.078262 (+17.48%)	0.168465 (+3.26%)	0.434168 (+0.20%)
Adam	0.068697 (+3.12%)	0.166362 (+1.97%)	0.433727 (+0.09%)

Table 7: Homomorphism densities of C_7 in graphons, which were found by our method and discretisation or by other baselines. The edge density constraint is specified by $p \in (0, 1)$.

	$p = 0.6$	$p = 0.7$	$p = 11/13$
Conjectured	0.0262613	0.0819155	0.31054965
Ours	**0.0262873 (+0.10%)**	**0.081972 (+0.07%)**	0.310558 (+0.0027%)
Hand-crafted	0.027556 (+4.93%)	0.082241 (+0.40%)	**0.310553 (+0.0012%)**
GD	0.028209 (+7.42%)	0.082752 (+1.02%)	0.310975 (+0.14%)
Adam	0.027994 (+6.60%)	0.082355 (+0.54%)	0.310562 (+0.0040%)

F Conjecture on Odd Cycles

In this section, we describe details about the conjecture about odd cycle density minimisation.

In Bennett et al. (2020), the optimal graphon's structure is constructed as follows. Let k be the integer satisfying $1 - \frac{1}{k} < p < 1 - \frac{1}{k+1}$. First, we divide the interval $[0, 1]$ to $(k + 1)$ sets, $X_1, \ldots, X_{k-1}, Y_1, Y_2$. We set the length of $X_1, \ldots, X_{k-1}$ equal to x, and set length of Y_1, Y_2 as $\frac{y}{2}$, so that $y = 1 - (k - 1)x$. Then we construct the graphon as follows:

$$W(x_1, x_2) = \begin{cases} 1 & \text{If } x_1 \in X_m, x_2 \in X_n, m \neq n \\ 1 & \text{If } x_1 \in X_m, x_2 \in Y_n, \text{ or conversely} \\ 0 & \text{If } x_1 \in X_m, x_2 \in X_m, \\ \rho & \text{If } x_1 \in Y_1, x_2 \in Y_2 \text{ or conversely} \\ 0 & \text{If } x_1 \in Y_m, x_2 \in Y_m, \end{cases}$$

where ρ is set accordingly to satisfy the edge density constraint.

Table 8: Homomorphism densities of the Petersen graph in graphons, which were found by our method or other baselines. The edge density constraint is specified by $p \in (0, 1)$.

	$p = 0.6$	$p = 0.7$	$p = 0.82$
Ours	**2.198922E-4 ± 5.23E-8**	**4.12332E-3 ± 2.68E-6**	**0.05006111 ± 9.86E-6**
Hand-crafted	3.9174E-4	4.3909E-3	0.050224
GD	5.354896E-4	5.22357E-3	0.05126811
Adam	2.199472E-4	4.12872E-3	0.05098772

Table 9: Homomorphism densities of the complete graphs in graphons, which were found by our method for the task of minimising $(t(K_r, W)^{\binom{s}{2}}) - t(K_s, W)^{\binom{r}{2}}$ over W.

	$r = 3, s = 4$	$r = 3, s = 5$	$r = 4, s = 5$
$t(K_r, W)$	0.538239	0.870075	0.783753
$t(K_s, W)$	0.437601	0.792634	0.737119
$t(K_r, W)^{\binom{s}{2}}$	0.02431687	0.24863651	0.08745706
$t(K_s, W)^{\binom{r}{2}}$	0.08379797	0.49798722	0.16040811
$t(K_r, W)^{\binom{s}{2}} - t(K_s, W)^{\binom{r}{2}}$	-0.05948110	-0.24935071	-0.07295104

One can then compute the C_5 density of this graphon, which is a polynomial $f_5(x, y, \rho)$ defined as following:

$$
\begin{aligned}
& f_5(x, y, \rho) \\
= & \left(\frac{(k-1)_5}{10} + \frac{(k-1)_4}{2} + \frac{(k-1)_3}{2} \right) x^5 \\
+ & \left(\frac{(k-1)_4}{2} + \frac{3(k-1)_3}{2} + \frac{(k-1)_2}{2} \right) x^4 y \\
+ & \left(\left(\frac{1}{2} + \frac{\rho}{2} \right) (k-1)_3 + \left(1 + \frac{\rho}{2} \right) (k-1)_2 \right) x^3 y^2 \\
+ & \left(\left(\frac{\rho}{2} + \frac{\rho^2}{2} \right) (k-1)_2 + \frac{\rho}{2}(k-1) \right) x^2 y^3 \\
+ & \frac{\rho^3}{2}(k-1) x y^4,
\end{aligned}
$$

where $(k-1)_r$ is a falling factorial, $(k-1)_r = (k-1) \cdot (k-2) \cdots (k-r)$. With these definitions, the optimal value is determined by optimising the value of x, y, ρ. In other words, one can obtain the conjectured optimal value by solving the following optimisation problem:

$$
\begin{aligned}
& \min f_5(x, y, \rho) \\
\text{w.r.t.} & (k-1)x + y = 1, \\
& (k-1)_2 x^2 + 2(k-1)xy + \rho y^3 = p, \\
& x, y \geq 0, \quad 0 \leq \rho \leq 0.5.
\end{aligned}
$$

Using the same structure, we can generalise the conjecture to arbitrary odd cycles C_{2k+1}, by replacing f_5 with to corresponding f_{2k+1}. For instance, f_7 is given as follows.

Table 10: Homomorphism densities of the triangles in graphons, which were found by our method and other variants. The edge density constraint is specified by $p = 7/9$.

	$t(K_3, W)$
Ours	**0.427215**
ResNet	0.428968
Constant-s	0.428190
Regulariser	0.434445
Fourth variant with a low learning rate	0.430071
Fourth variant with a high learning rate	0.431645

$$
\begin{aligned}
&f_7(x, y, \rho) \\
&= \left(\frac{(k-1)_7}{14} + (k-1)_6 + 4(k-1)_5 + 5(k-1)_4 + \frac{3(k-1)_3}{2} \right) x^7 \\
&+ \left(\frac{(k-1)_6}{2} + 5(k-1)_5 + \frac{25(k-1)_4}{2} + \frac{15(k-1)_3}{2} + \frac{(k-1)_2}{2} \right) x^6 y \\
&+ \left((k-1)_5 + (3\rho+7)(k-1)_4 + \left(\frac{7\rho}{2} + 10 \right) (k-1)_3 + \left(\frac{\rho}{2} + 2 \right) (k-1)_2 \right) x^5 y^2 \\
&+ \left(\left(\frac{3\rho}{2} + \frac{1}{2} \right) (k-1)_4 + \left(\frac{3\rho^2}{2} + 6\rho + \frac{5}{2} \right) (k-1)_3 + \left(\frac{\rho^2}{2} + 3\rho + 2 \right) (k-1)_2 \right) x^4 y^3 \\
&+ \left(\left(\frac{\rho^3}{2} + \frac{3\rho^2}{2} + \frac{\rho}{2} \right) (k-1)_3 + \left(\frac{\rho^3}{2} + 3\rho^2 + \frac{3\rho}{2} \right) (k-1)_2 + \frac{\rho}{2}(k-1) \right) x^3 y^4 \\
&+ \left(\left(\frac{\rho^4}{2} + \rho^3 \right) (k-1)_2 + \rho^3 (k-1) \right) x^2 y^5 \\
&+ \frac{\rho^5}{2} (k-1) x y^6.
\end{aligned}
$$

G Details of Hand-crafted Graphons

In this section, we describe the hand-crafted graphons we used as the baseline of our experiments. We will consider the densities satisfying $1 - 1/k < p < 1 - 1/(k+1)$, where k is a positive integer.

The first two instances are the graphons constructed for the optimal solution of edge density constrained K_3 homomorphism density minimisation. We construct a complete $(k+1)$-partite graphon, i.e., we divide the interval $[0, 1]$ to $(k+1)$ groups, $X_1, \ldots, X_{k+1}$, and define

$$
W_i(x_1, x_2) = \begin{cases} 1 & \text{If } x_1 \in X_m, x_2 \in X_n, m \neq n \\ 0 & \text{If } x_1 \in X_m, x_2 \in X_m. \end{cases}
$$

We set $|X_1| = \cdots = |X_k| = x$ and $|X_{k+1}| = 1 - kx$, and choose x so that the edge density constraint is satisfied, which can be obtained by solving the quadratic equation

$$
k(k-1)x^2 + 2kx(1-kx) = p.
$$

Solving this quadratic equation gives two solutions, one with $x > 1 - kx$ (W_1) and the other with $x < 1 - kx$ (W_2), and the first solution gives the optimal solution for K_3 density minimisation.

The second instance is constructed by deleting edges from the complete k-partite graphon. We divide the interval $[0, 1]$ to k groups $X_1, \ldots, X_k$ with $|X_1| = \cdots = |X_k| = 1/k$, and define

$$
W_3(x_1, x_2) = \begin{cases} \frac{kp}{k-1} & \text{If } x_1 \in X_m, x_2 \in X_n, m \neq n \\ 0 & \text{If } x_1 \in X_m, x_2 \in X_m, \end{cases}
$$

$$
W_4(x_1, x_2) = \begin{cases} \frac{k(k+1) - p(k+1)^2}{2} & \text{if } x_1 \in X_1, x_2 \in X_2 \text{ or conversely} \\ 0 & \text{if } x_1 \in X_m, x_2 \in X_m, \\ 1 & \text{otherwise.} \end{cases}
$$

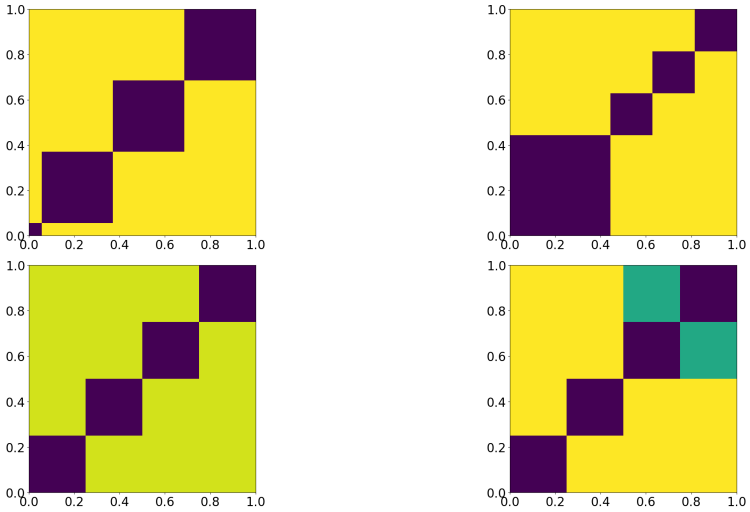


Figure 9: Hand-crafted graphons W_1 (top left), W_2 (top right), W_3 (bottom left), and W_4 (bottom right) with edge density $p = 0.7$.

Finally, the last graphon is the constant graphon $W(x_1, x_2) = p$.

We visualised $W_1, \ldots, W_4$ for $p = 0.7$ in Fig. 9.