Gaussian Processes on Hypergraphs

Thomas Pinder[1], Kathryn Turnbull[1]
Christopher Nemeth[1], David Leslie[1]

[1]Department of Mathematics and Statistics, Lancaster University, UK

June 4, 2021

Abstract

We derive a Matérn Gaussian process (GP) on the vertices of a hypergraph. This enables estimation of regression models of observed or latent values associated with the vertices, in which the correlation and uncertainty estimates are informed by the hypergraph structure. We further present a framework for embedding the vertices of a hypergraph into a latent space using the hypergraph GP. Finally, we provide a scheme for identifying a small number of representative inducing vertices that enables scalable inference through sparse GPs. We demonstrate the utility of our framework on three challenging real-world problems that concern multi-class classification for the political party affiliation of legislators on the basis of voting behaviour, probabilistic matrix factorisation of movie reviews, and embedding a hypergraph of animals into a low-dimensional latent space.

1 Introduction

Gaussian processes (GPs) are a popular type of stochastic process commonly used in machine learning for modeling distributions over real-valued functions (Rasmussen and Williams, 2006). In recent years, GPs have been successfully used in areas such as optimisation (Mockus, 2012), variance-reduction (Oates et al., 2017) and robotics (Deisenroth et al., 2015). This recent blossoming is due to the GP's ability to model complex functional relationships with accompanying well-calibrated uncertainty estimates.

Careful consideration of the underlying data's representation has facilitated more expressive GP priors to be constructed. Examples of this include the modelling of regular time-series models (Särkkä and Solin, 2019; Adam et al., 2020), spatial data observed on a uniform grid (Solin and Särkkä, 2019) and multivariate outputs where inter-variable correlations exists (Alvarez et al., 2012). In this

work, we develop and implement a GP prior for situations where relationships between the explanatory variables in the data admit a *hypergraph* structure.

As a motivating example, consider modelling the party affiliation of a group of politicians using legislative information. It is natural to expect that politicians who collaborate on a piece of legislation share some similarity. Since any set of politicians may work together, this dependence can be represented by a hypergraph where each politician is a vertex and each hyperedge corresponds to a piece of legislation. Whilst we may also describe this data structure as a graph of pairwise collaborations, this representation cannot express the full structural information encoded in the hypergraph where collaborations occur between two, or more, politicians. When the relationship between observations is represented by a hypergraph, the Euclidean measure of distance between pairs of observations is no longer appropriate and we must instead build a distance function using the structure of the underlying hypergraph.

The contributions of this paper can be seen as a generalisation of Borovitskiy et al. (2020a) whereby we introduce a hypergraph kernel to allow higher-order interactions to be represented in the GP prior. This results in a prior distribution that is more representative of the underlying data's structure, and we find that it leads to significant improvements in the GP posterior's inferential performance. In addition to this, we also present a novel approach for embedding hypergraphs in a lower dimensional latent space through a Gaussian process latent variable model (GPLVM), present a new scheme for selecting inducing points in a hypergraph GP and finally show how our hypergraph GP can be used within a probabilistic matrix factorisation (PMF) framework.

2 Background

2.1 Gaussian processes

For a set $\mathcal{X}$, a stochastic process $f : \mathcal{X} \to \mathbb{R}$ is a GP if for any finite collection $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\} \subset \mathcal{X}$ the random variable $\mathbf{f} := f(X)$ is distributed jointly Gaussian (Rasmussen and Williams, 2006). We write a GP $f \sim \mathcal{GP}(\mu, k)$ with mean function $\mu : \mathcal{X} \to \mathbb{R}$ and $\boldsymbol{\theta}$-parameterised kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ where k gives rise to the Gram matrix $\mathbf{K_{xx}}$ such that the $(i, j)^{\text{th}}$ entry is computed by $[\mathbf{K_{xx}}]_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$. In standard expositions of stationary Gaussian processes, with $\mathcal{X} = \mathbb{R}^d$, $k(\mathbf{x}_i, \mathbf{x}_j)$ is a function of the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$, whereas this article introduces a kernel that is evaluated on vertices of a hypergraph. Without loss of generality, we assume $\mu(\mathbf{x}) := 0$ for all $\mathbf{x}$ and drop dependency on $\boldsymbol{\theta}$ for notational convenience.

Assume an observed dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$ of N training example pairs with inputs $\mathbf{x}_i \in \mathcal{X}$ and outputs $y_i \in \mathbb{R}$ distributed according to $y \,|\, \mathbf{f} \sim \prod_{i=1}^N p(y_i \,|\, f_i)$. Then the predictive density for latent function values $\mathbf{f}_\star$ at a finite collection of

$N^\star$ test points $X^\star = \{\mathbf{x}_1^\star, \ldots, \mathbf{x}_{N^\star}^\star\}$ is given by

$$p(\mathbf{f}_\star \,|\, \mathcal{D}) = \int p\left(\mathbf{f}_\star, \mathbf{f} \,|\, \mathcal{D}\right) \mathrm{d}\mathbf{f} = \int p\left(\mathbf{f}_\star | \mathbf{f}\right) p(\mathbf{f}|\mathcal{D}) \,\mathrm{d}\mathbf{f}, \tag{1}$$

where $p(\mathbf{f}|\mathcal{D}) \propto p(\mathbf{y}|\mathbf{f})p(\mathbf{f})$ is the posterior density of $\mathbf{f}$ given the training data. When the observations are distributed according to a Gaussian likelihood, both $p(\mathbf{f}_\star \,|\, \mathbf{f})$ and $p(\mathbf{f} \,|\, \mathcal{D})$ are Gaussian densities. The marginal distribution of a Gaussian is also Gaussian and so the posterior in Equation (1) is a Gaussian distribution with a closed-form expression for the mean and covariance. When the likelihood function is non-Gaussian, the posterior distribution over the latent function $p(\mathbf{f} \,|\, \mathcal{D})$ has no closed-form expression and inference for Equation (1) is no longer tractable. Popular solutions that address this problem involve approximating $p(\mathbf{f} \,|\, \mathcal{D})$ through a Laplace approximation (Williams and Barber, 1998) or variational inference (Opper and Archambeau, 2008). Alternatively, the latent function's values can be inferred directly using either Markov chain Monte-Carlo (MCMC) (Murray et al., 2010) or Stein variational gradient descent (Pinder et al., 2020).

The primary cost incurred when computing the GP posterior is the inversion of the Gram matrix $\mathbf{K_{xx}}$. Sparse approximations (Snelson and Ghahramani, 2005) temper this cubic in N cost through the introduction of a set of J *inducing points* $Z = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_J\} \in \mathcal{X}$ and corresponding function outputs $\mathbf{u} = f(Z)$. Letting $J \ll N$, this approach reduces the computational requirement to $\mathcal{O}(NJ^2)$ from $\mathcal{O}(N^3)$. Following Titsias (2009), we augment the posterior distribution with $\mathbf{u}$ to give $p(\mathbf{f}, \mathbf{u} \,|\, \mathbf{y}) = p(\mathbf{f} \,|\, \mathbf{u}, \mathbf{y})p(\mathbf{u} \,|\, \mathbf{y})$ and then introduce the variational approximation $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \,|\, \mathbf{u})q(\mathbf{u})$. Constraining q to be a multivariate Gaussian $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, S)$ and collecting our parameters of interest $\psi = \{Z, \mathbf{m}, S, \boldsymbol{\theta}\}$, we can determine the *optimal* parameters $\psi^\star$ for the variational approximation as

$$\psi^\star = \underset{\psi \in \Psi}{\arg\min} \, \mathrm{KL}(q(\mathbf{f}, \mathbf{u}) || p(\mathbf{f}, \mathbf{u} \,|\, \mathbf{y}))$$

$$\geq \underset{\psi \in \Psi}{\arg\max} \sum_{n=1}^{N} \mathbb{E}_{q(\mathbf{u}, \mathbf{f})} \left[\log p(y_i \,|\, f_i)\right] - \mathrm{KL}(q(\mathbf{u}) || p(\mathbf{u})) \tag{2}$$

where Ψ is the space of all possible parameters and $p(\mathbf{u}) = \mathcal{N}(\mathbf{u} \,|\, \mathbf{0}, \mathbf{K_{zz}})$ such that $[\mathbf{K_{zz}}]_{i,j} = k(\mathbf{z}_i, \mathbf{z}_j)$. The quantity in Equation (2) is commonly referred to as the evidence lower bound (ELBO) and is equal to the marginal log-likelihood when $\mathrm{KL}(q(\mathbf{u})||p(\mathbf{u})) = 0$. For a full introduction to variational sparse GPs see Leibfried et al. (2020).

2.2 Hypergraphs

A *hypergraph* (Bretto, 2013) is a generalisation of a graph in which interactions occur among an arbitrary set of nodes (see Figure 1). This type of data provides

a flexible and descriptive framework to encode higher-order relationships within a graph-like structure. Formally, we let $\mathcal{G} = \{V, E\}$ be a hypergraph with vertices $V = \{v_1, \ldots, v_N\}$ and *hyperedges* $E = \{e_1, \ldots, e_M\}$. Each hyperedge $e_i \in E$ corresponds to a subset of vertices from V with no repeated elements. A hyperedge e_i is said to be *weighted* when it has a positive value $w(e_i)$ assigned to it and *incident* to a vertex v_j if $v_j \in e_i$. We represent a hypergraph via a $|N| \times |M|$ incidence matrix H with $(j, i)^{\text{th}}$ entry equal to 1 if e_i is incident to v_j and 0 otherwise.

Every hypergraph also admits a bipartite representation in which each hyperedge is assigned a node, and an edge from a node vertex to a hyperedge vertex indicates incidence (see Figure 1). Since a hypergraph generalises the graph representation, it follows that these structures coincide when each hyperedge contains precisely two nodes. Whilst we may obtain a graph from a given hypergraph (see Appendix A.2), this inevitably loses structural information which cannot be recovered.
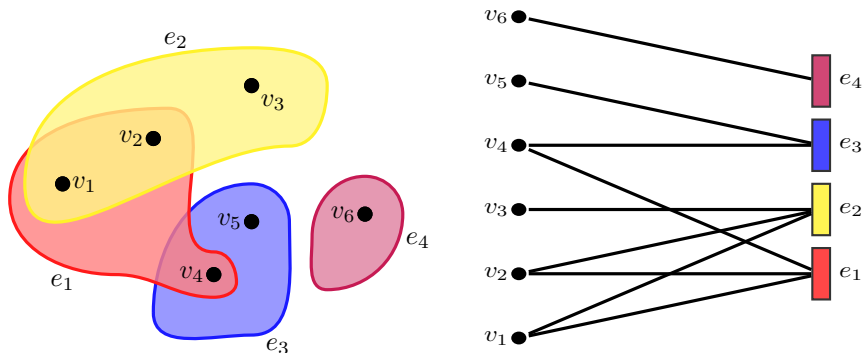

Figure 1: A hypergraph comprised of four hyperedges among six vertices and the corresponding bipartite representation. See Section 2.1.2 in Bretto (2013) for details of this relationship.

The Laplacian of a hypergraph is an important matrix representation which can be used to study hypergraph properties. This matrix has proven useful as part of machine learning algorithms, such as spectral clustering, and, due to its importance, several authors have proposed analogous hypergraph Laplacians (see Agarwal et al. (2006), Battiston et al. (2020) for examples). We rely on the hypergraph Laplacian proposed by Zhou et al. (2006) for spectral clustering, which is given by

$$\Delta = I - D_v^{-1/2} H D_e^{-1} H^\top D_v^{-1/2}, \tag{3}$$

where D_v and D_e are diagonal matrices with non-zero entries containing the node and hyperedge degrees, respectively. We write

$$[D_v]_{i,i} = \sum_{e \in E} w(e) h(v_i, e) \qquad [D_e]_{i,i} = \sum_{v \in V} h(v, e_i), \tag{4}$$

4

where $h(v, e) = 1$ if v is incident to e i.e., $v \in e$. In this work we only considered unweighted hypergraphs (i.e. $w(e_i) = 1$ for $1 \leq i \leq M$) however extending our framework to accommodate weighted hyperedges poses no challenge.

3 Gaussian process on hypergraphs

We wish to define a GP prior for functions over the space containing the vertices of a hypergraph i.e., $\mathcal{X} = V$. We do so by building from a stochastic partial differential equation (SPDE) formulation of a GP prior over Euclidean spaces.

SPDE kernel representations Defining kernel functions in terms of an SPDE provides a flexible framework that facilitates the derivation of kernels on non-Euclidean spaces. To see this, we consider the SPDE

$$\mathcal{T}f = \mathcal{W} \tag{5}$$

where $\mathcal{T} : \mathcal{X} \to \mathcal{H}$ is a bounded linear operator on a Hilbert space $\mathcal{H}$, f is a zero-mean Gaussian random field and $\mathcal{W}$ a white noise process (Lototsky and Rozovsky, 2017). For a Euclidean domain i.e., $\mathcal{X} = \mathbb{R}^d$, Whittle (1963) has shown that the Matérn kernel satisfies Equation (5) through

$$\left(\frac{2\nu}{\ell^2} + L \right)^{\nu/2 + d/4} f = \mathcal{W}, \tag{6}$$

where L is the Laplace operator $Lf(\mathbf{x}) = \nabla \cdot \nabla f(\mathbf{x})$; the divergence of the gradient of f, and $\nu \in \mathbb{R}_{>0}$ and $\ell \in \mathbb{R}_{>0}$ are the smoothness and lengthscale parameters of the Matérn kernel respectively.

Hypergraph domain We transfer the construction Equation (6) into the hypergraph domain by taking $\mathcal{W}$ to be a multivariate spherical Gaussian and replacing the Laplace operator L with the hypergraph Laplacian Δ. (We can also drop the term $d/4$ in the exponent, since it is not required for smoothness concerns in the discrete space $\mathcal{X}$.) Since we have a discrete space $\mathcal{X}$, Equation (6) becomes

$$\left(\frac{2\nu}{\ell^2} + \Delta \right)^{\nu/2} \mathbf{f} = \mathcal{W}, \tag{7}$$

where $\mathbf{f}$ is the vector of function values at the vertices of the hypergraph. We can rearrange Equation (7) to give the hypergraph GP prior

$$\mathbf{f} \sim \mathcal{N} \left(0, \left(\frac{2\nu}{\ell^2} + \Delta \right)^{-\nu} \right). \tag{8}$$

Hypergraph dual For a given hypergraph, we can construct a hypergraph dual whose vertices correspond to the hyperedges in the original hypergraph. This relationship is made clear through the bipartite representation in Figure 1. We can leverage this duality to facilitate inference on either the vertices or hyperedges in our initial hypergraph, and we make use of this feature in Section 5.3.

Connection to graph GPs If we consider hypergraphs of order 2, i.e. $|e| = 2$ for all $e \in E$, then we recover the GP formulated in Borovitskiy et al. (2020a) as a special case of the hypergraph GP.

3.1 Sparse hypergraph GPs

Revisiting the ELBO term in Equation (2), we can see that for GPs defined on continuous domains, the inducing points $Z = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_J\}$ are treated as a model parameter that we optimise. Despite this, poor initialisation can lead to significantly slower convergence and often a poor variational approximation (Burt et al., 2020). For hypergraphs, such an optimisation is not possible as the notion of an inducing point is now a discrete quantity that corresponds to a specific vertex's index in the hypergraph. We hereafter refer to inducing points in a hypergraph as *inducing vertices*. As we cannot optimise our inducing vertices, it is important that we initialise them effectively as a poor initialisation cannot be rectified through an optimisation scheme.

Our approach is a three-stage process where we first assign each vertex a globally measured *importance* score $\gamma \in [0, 1]$. We use this measure to ensure that the most influential vertices in our hypergraph are present in the inducing set. Second, we cluster the vertices of the hypergraphs by using k-means clustering on the Laplacian matrix's eigenvalues. Finally, using the clustered vertices, we first sample a cluster with probability equal to the the cluster's normalised size, then select the vertex with the largest importance score in the respective cluster.

Vertex importance To compute importance scores for each vertex we project our hypergraph onto a graph and calculate the eigencentrality (see Kolaczyk, 2009) of each vertex. We first calculate the adjacency matrix $A_v = HH^\top$ with diagonal entries equal to the vertex degrees and off-diagonals equal to the number of times two vertices are incident to one another. If we now define the stochastic matrix $Q = A_v D_v^{-1}$ that scales the adjacency matrix by the vertices' degree, then we can obtain centrality measures through $\lambda\boldsymbol{\gamma} = Q\boldsymbol{\gamma}$. We can determine $\boldsymbol{\gamma}$ by identifying the eigenvector that corresponds to the largest eigenvalue λ, and the centrality score for the vth vertex is then given by γ_v. We note that H and D_v are non-negative matrices and therefore Q is also non-negative. Consequently, the Perron-Frobenius theorem guarantees the existence of a unique $\boldsymbol{\gamma}$.

Clustering To cluster the vertices of our hypergraph, k-means clustering is carried out on the eigenvectors of the hypergraph's Laplacian matrix Equation (3). In the absence of informative prior information, such as the number of observation classes, we suggest selecting a conservative value of $k \ll J$.

Inducing vertex selection A set of J inducing vertices can now be identified by a two-step procedure. Let k_i denote the ith cluster that contains $|k_i|$ vertices and cat be the categorical distribution with k categories with corresponding probabilities $\mathbf{p} = \{p_1, p_2, \ldots, p_k\}$. The jth element of Z can then be selected by

$$\text{Sample: } s \sim \text{cat}(\mathbf{p}), \text{ where } \mathbf{p} \propto \left\{ \frac{|k_1|}{N}, \frac{|k_2|}{N}, \ldots, \frac{|k_s|}{N} \right\}$$

$$\text{Select: } z_j = \underset{v \in s \,|\, v \notin Z_{1:(j-1)}}{\arg \max} \gamma_v. \tag{9}$$

This process is repeated until $|Z| = J$.

Unlike uniform random sampling, the approach outlined here is more robust to situations where a class imbalance in the observations' labels is present due to the clustering step performed. Further, we are able to identify the most influential vertices within the graph through the assignment of an eigen-based importance measure to each vertex; a metric which captures a vertex's *global* importance. Despite this, the framework presented here is highly modular, and the practitioner is free to use alternative clustering methods, such as the popular DBSCAN (Ester et al., 1996), or alternative centrality measures on the hypergraph's vertices (Benson, 2019).

3.2 Hypergraph embedding

The task of embedding the vertices of a hypergraph into a lower-dimensional *latent space* is a common approach to understanding the relational structure within a hypergraph (Zhou et al., 2006; Turnbull et al., 2019). Here we establish a connection between the Gaussian process latent variable model (GPLVM) (Lawrence, 2003) and the Matérn hypergraph GP Equation (8), and demonstrate how this synergy can be leveraged to infer a hypergraph latent space embedding in a computationally-efficient manner.

Working under the assumption that our hypergraph can be embedded in a low-dimension *latent space*, the aim here is to learn the position of each vertex in the latent space through a GP. We let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, represent our Q-dimensional latent space, such that the ith vertex's latent space coordinate is given by $\mathbf{x}_i \in \mathbb{R}^Q$.

We wish to learn the posterior distribution of the vertices' latent positions conditional on the hypergraph incidence matrix H, i.e. $p(X|H) \propto p(H|X)p(X)$. We can use our hypergraph Gaussian process model (Section 3) to create a

mapping between the latent positions and the hypergraph structure using the following model,

$$p(H \mid X)p(X) = \left(\int p(H \mid F)p(F \mid X)\mathrm{d}F \right) p(X), \qquad (10)$$

where F is an unobserved matrix with the same dimensions as the incidence matrix H, with each column f_j sampled independently from a GP prior, such that

$$p(H \mid F) = \mathcal{N}(F, \sigma^2 I_N), \quad p(f_j \mid X) = \mathcal{N}(\mathbf{0}, \mathbf{K_{xx}}\mathbf{K}_{VV}) \text{ and } p(X) = \mathcal{N}(\mathbf{0}, I_N).$$

The Gram matrix $\mathbf{K_{xx}}$ is calculated using a Euclidean kernel, e.g., the squared exponential, on the domain $\mathbb{R}^Q \times \mathbb{R}^Q$, I_N is a $N \times N$ identity matrix, and $\mathbf{K}_{VV}$ is the hypergraph kernel from Equation (8), which uses the hypergraph Laplacian to inform the prior model in an empirical Bayes approach analogous to using a spatial variogram to fix the kernel parameters in a Euclidean model e.g., Diggle et al. (1998). The incidence matrix H is a binary matrix, however, in this section we make the assumption that $H|F$ follows a Gaussian distribution. This assumption allows us to solve the integral Equation (10) analytically,

$$H|X \sim \mathcal{N}(0, \mathbf{K_{xx}}\mathbf{K}_{VV} + \sigma^2 I_N).$$

Without this assumption, approximate inference techniques such as MCMC or variational inference would be required.

The model described here is incredibly flexible as it enables the practitioner to posit their beliefs around the *smoothness* of the latent space through the ν parameter in Equation (8). Larger values of ν will result in a more dispersed latent space and larger degrees of separation will be enforced between clusters. Further, covariate information at the vertex level can easily be incorporated into the embedding function by specifying a second kernel that acts across the covariate space of the vertex set and then computing the product of this covariate kernel with the base graph kernel.

4 Related work

To the best of our knowledge, this is the first work to consider GP inference for hypergraphs. However, the SPDE representation of a Matérn kernel has been successfully used by Lindgren et al. (2011) to allow for highly scalable inference on Gaussian Markov random fields (GMRFs) (Rue and Held, 2005) and more recently Riemannian manifolds (Borovitskiy et al., 2020b). Parallel to this line of work, (Zhi et al., 2020) derive graph GP priors by extending the multi-output GP prior given in van der Wilk et al. (2020). Finally, Opolka and Liò (2020) derive a deep GP prior (Damianou and Lawrence, 2013) that acts on a vertex set using the spectral convolution of a graph's adjacency matrix, as

given by Kipf and Welling (2017). Inference on graphs using GPs, as shown in the aforementioned works, is effective. However, as we show in Section 5, such models can be greatly improved by utilising the full hypergraph structure.

Within the deep learning community, graph neural networks (GNNs) (Bronstein et al., 2016) have been studied in the context of hypergraphs from both an algorithmic (Feng et al., 2019) and theoretical (Bodnar et al., 2021) perspective. More recently, hypergraph GNN architectures have been extended to incorporate attention modules (Bai et al., 2021). Unfortunately, GNNs are currently incapable of quantifying the predictive uncertainty on the vertices of a hypergraph. In contrast, the hypergraph GP presented here is able to produce well-calibrated uncertainty estimates (see Section 5.1).

Analysis of hypergraph data appears much more broadly in the statistics and machine learning literature (see Battiston et al. (2020) for contemporary review), and typical tasks involving hypergraphs include label prediction, classification and clustering (Gao et al., 2020). Within the GP, the hypergraph informs the dependence structure among the variables associated with the vertices and we note that this departs from a common hypergraph modelling set-up in which there is uncertainty on the hyperedges. Finally, whilst we rely on the hypergraph Laplacian proposed in Zhou et al. (2006), we note the existence of other Laplacians which may be used in our context (for example, see Chung (1993); Hu and Qi (2015); Saito et al. (2018) for examples), depending on the suitability of the assumptions for a given hypergraph.

5   Experiments

In this section we illustrate the variety of modelling challenges that can be addressed through a hypergraph Gaussian process. A full description of the experimental set up used below can be found in Appendix A.1. Further, we release a Python package that enables GP inference on hypergraphs using GPFlow[1] (de G. Matthews et al., 2017) and HyperNetX (HyperNetX, since 2019) along with code to replicate the experiments at `https://github.com/RedactedForReview`.

5.1   Multi-class classification on legislation networks

In this first section we demonstrate the additional utility that is given from a hypergraph representation by comparing our proposed model to its simpler graph analogue. To achieve this, we consider data that describes the co-sponsorship of legislation within the Congress of the Republic of Peru in 2007 (Lee et al., 2017). To represent this dataset as a hypergraph, we form a hyperedge for each piece of legislation and the constituent vertices correspond to the members of Congress responsible for drafting the respective legislation. In the 2007 Congress there

---

[1]Licensed under Apache 2.0.

were three distinct groups[2]: the government, opposition and minority groups. Our task here is to predict the group affiliation of a member of Congress given only the hypergraph structure.

Table 1: The performance of the GP models from Section 5.1 on a graph and hypergraph structure. Results are reported for the 40 held-out vertices, split by the underlying (hyper)graph representation. Bold values denote the best performing model and standard errors are computed across 10 random partitions of the data. For every metric, excluding expected calibration error (ECE), a larger value is better.

Metric	Hypergraph	Graph Binary expansion	Graph Weighted expansion
Accuracy	$\mathbf{0.9 \pm 0.002}$	0.7 ± 0.015	0.75 ± 0.009
Recall	$\mathbf{0.89 \pm 0.002}$	0.74 ± 0.002	0.78 ± 0.006
Precision	$\mathbf{0.9 \pm 0.002}$	0.73 ± 0.033	0.79 ± 0.007
ECE	$\mathbf{0.13 \pm 0.001}$	0.29 ± 0.003	0.2 ± 0.002
Log-posterior density	$\mathbf{-0.34 \pm 0.017}$	-0.89 ± 0.001	-0.45 ± 0.005

As far as we are aware, this is the first piece of work to consider GP modelling on hypergraphs, so no directly comparative method is available. Therefore, to establish suitable benchmarks, we represent our hypergraph as a graph using both a weighted and binary clique expansion (Agarwal et al., 2005); a process we briefly outline in Appendix A.2. From thereon, GP modelling can be accomplished using the graph kernel provided in Borovitskiy et al. (2020a). Although additional mappings from a hypergraph to a graph are commonly used in the literature, such as the star expansion, we focus on the clique expansion since this preserves the roles of the vertices.

We employ the categorical likelihood function of Hernández-Lobato et al. (2011) and a multi-output GP $f : V \to \mathbb{P}^2$, where $\mathbb{P}^2$ is the probability 2-simplex and each output dimension corresponds to the probability of the respective vertex being attributed to one of the three political groups. As can be seen in Table 1, the hypergraph representation yields significantly fewer misclassified nodes with equally compelling precision and recall statistics. GPs are commonly used due to their ability to quantify predictive uncertainty and, as can be seen by the expected calibration error (ECE) values in Table 1, the hypergraph representation facilitates substantially improved posterior calibration.

In addition to the universally superior predictive measures reported in Table 1, the ELBO curves in Figure 2 show that the hypergraph representation facilitates more efficient optimisation. This result is particularly noteworthy as a common criticism of GPs is the computational demands invoked by their optimisation. Therefore, being able to achieve more efficient optimisation is critically important if we hope for these GP models to be more widely adopted by machine learning

---

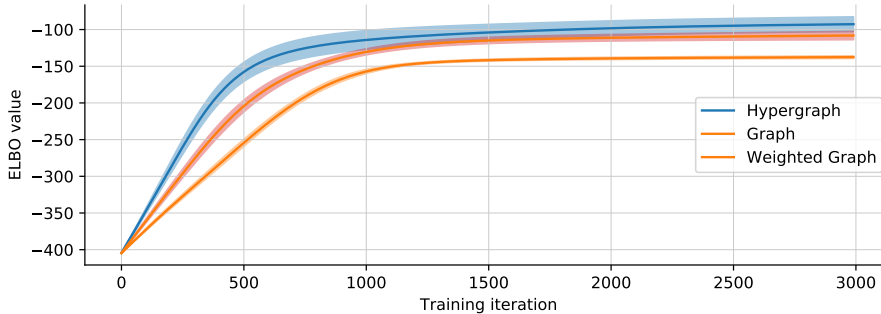[2]Groups in the Republic of Peru's Congress are collections of political parties

Figure 2: Convergence of the ELBO throughout optimisation. A larger value is better. Maximising the ELBO is analogous to minimising the Kullback-Leibler divergence from our approximate GP to the true GP. We can see that representing the data as a hypergraph in our GP yields substantially more efficient optimisation, and the resultant model gives a tighter bound on the true log-likelihood.

practitioners.

5.2 Hypergraph latent embeddings

In this example we demonstrate the enhanced utility provided by the hypergraph GPLVM outlined in Section 3.2 in comparison to traditional spectral models that exist in the hypergraph literature, namely the work of Zhou et al. (2006). The dataset under consideration is the Zoo data from the UCI Machine Learning Depository (Dua and Graff, 2017) and our goal is to embed the animals' relational structure into a 2-dimensional latent space. We represent the dataset as a hypergraph, such that each vertex corresponds to a specific animal, and each hyperedge corresponds to an animal's attribute e.g., number of legs, presence of a tail. There are seven possible labels within the dataset that roughly describe an animal's taxonomic class. We remove this attribute from the dataset and only use it for the colouring of the vertices' latent space positions in Figure 3. We use a squared exponential kernel to compute the Gram matrix over the latent spaces' coordinates (denoted $\mathbf{K_{xx}}$ in Section 3.2).

We can visually see from Figure 3 that the hypergraph GPLVM is better able to uncover the latent group structure that is present in the data through the more homogeneous group clustering. Further, using the convex hulls in Figure 3 we are able to measure the degree to which each latent space representation has been able to recover the original classes. We report the adjusted mutual information (Nguyen et al., 2009), homogeneity and completeness (Rosenberg and Hirschberg, 2007) in Table 2 where it can be seen that our hypergraph GPLVM outperforms the approach of Zhou et al. (2006) for all considered metrics.
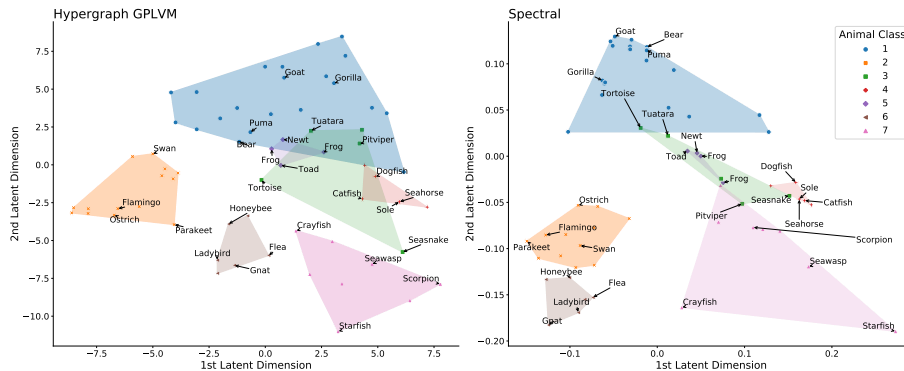
Figure 3: Visualisation of the latent space projections learned using our hypergraph based GPLVM and the method outlined in Zhou et al. (2006) titled *spectral*. Observations are coloured based upon their respective class and the shadings denote the convex hull of the entire class. To avoid a cluttered layout, we randomly label four animals per class and provide fully labelled plot in Appendix B.1.

Table 2: Adjusted mutual information, homogeneity and completeness of the latent spaces produced by our proposed hypergraph GPLVM and the spectral embedding of Zhou et al. (2006). Each score is defined on the domain $[0, 1]$ and a higher score is better. A description of these three measures is provided in Appendix A.

	Hypergraph GPLVM	Spectral embedding
Adjusted mutual information	**0.689**	0.606
Homoegeneity	**0.774**	0.668
Completeness	**0.683**	0.633

5.3 Kernelised probabilistic matrix factorisation

In this section we study the effect of a hypergraph kernel when used for kernel probabilistic matrix factorisation (KPMF) (Zhou et al., 2012). Given a partially observed matrix $R \in \mathbb{R}^{N_U \times N_W}$, for example a matrix of movie ratings, our aim is to predict the missing values of R using the assumption that the outer product of two latent matrices $U \in \mathbb{R}^{N_U \times D}$ and $W \in \mathbb{R}^{N_W \times D}$ generates R where D is the latent factors' dimension. We place zero-mean GP priors over the columns of U and W,

$$U_{:,d} \sim \mathcal{GP}(0, \kappa_U), \qquad W_{:,d} \sim \mathcal{GP}(0, \kappa_W). \tag{11}$$

A factorised Gaussian likelihood is placed over R such that

$$p(R \mid U, W, \sigma_n^2) = \prod_{n=1}^{N} \prod_{m=1}^{M} \mathcal{N} \left(R_{n,m} \mid U_{n,:} W_{m,:}^{\top}, \sigma^2 \right)^{\delta_{n,m}} \tag{12}$$

where $\delta_{n,m} = 1$ if $R_{n,m}$ is not empty, and 0 otherwise. We then learn the optimal pair of latent matrices by optimising the model's log-posterior with respect to U and W.

To employ this model we use the MovieLens-100k dataset that consists of 100,000 reviews on 1682 movies from 943 users (Harper and Konstan, 2016). We define the GP priors in Equation (11) as our hypergraph prior from Equation (8). Letting $R \in \mathbb{R}^{943 \times 1682}$ be the matrix of movie ratings, we can construct the incidence matrix H of the hypergraph for which users represent vertices and the collections of users who review the same movie as hyperedges by calculating $[H]_{n,m} = \delta_{n,m}$ for $1 \leq n \leq N_U$ and $1 \leq m \leq N_W$ from Equation (12). Through the hypergraph's dual representation (see Section 3), we are able to define incidence matrices where movies are vertices and hyperedges contain the set of movies that an individual user reviewed by $H^{\top}$.

Table 3: Root-mean-square error (RMSE) for our hypergraph GP KPMF model in Section 5.3 and comparative diffusion model from Zhou et al. (2012). We report RMSE on the held-out data where a lower RMSE indicates a better model. All models are fit to datasets whereby 20% and 80% of the data is held back for testing. The sparse variant of our model uses an inducing point set with size equal to $^1\!/_{20}$th of the U and W matrices. Standard errors reported here are across 10 random partitions of the data and are significant after the third decimal place.

Testing proportion	Diffusion	Hypergraph KPMF Full	Hypergraph KPMF Sparse
20%	1.39 ± 0.0	$\mathbf{1.21 \pm 0.0}$	1.40 ± 0.1
80%	1.51 ± 0.0	$\mathbf{1.27 \pm 0.0}$	1.56 ± 0.1

In Table 3 we compare our hypergraph-based KPMF to the KPMF model given in Zhou et al. (2012) which uses a diffusion kernel (Kondor and Lafferty, 2002) that we describe fully in Appendix A.3. Further, we test the effect of our inducing point scheme described in Section 3.1 by approximating κ_U and κ_W with 45 and and 80 inducing vertices respectively. As can be seen by the smaller RMSE in Table 3, the hypergraph kernel offers substantial improvements in comparison to the regular KPMF model. Whilst attaining a larger RMSE than the dense hypergraph KPMF model, the sparse approximation is able to compute Equation (12) 4.3 times faster than the both diffusion and dense KPMF models. We note that a sparse approximation is not strictly necessary in this example, however, selecting a dataset where fitting full rank kernels is possible allows us to quantitatively assess the difference in predictive RMSEs. Furthermore, this

sparse approximation allows us to scale this model to datasets where U and/or W contains millions of observations. In such a setting, fitting a dense kernel would yield an intractable likelihood Equation (12) due to the requirement of inverting κ_U and κ_W. However, using our sparse kernel approximation would permit scaling to datasets of this size.

6   Concluding remarks

Limitations   Whilst we have found the hypergraph Laplacian from Equation (3) to perform well experimentally, we are aware of several criticisms of this in the literature (Agarwal et al., 2005; Ren et al., 2008). Despite the limitations this could cause, we see this as an illuminating potential pathway for future research through the consideration of alternative hypergraph Laplacians.

Finally, we only consider hypergraphs that are fixed in the number of nodes and hyperedges. Whilst valid for the use cases we consider in Section 5, there exists a large number of hypergraph applications and methodologies whereby the number of vertices and/or hyperedges is either evolving or considered unknown.

Final remarks   In this work we consider the task of performing inference on the vertices of a general hypergraph. To do so, we establish a connection with the Matérn graph kernel and provide a detailed framework for positing the rich structure of a hypergraph into this kernel. We further provide a framework for embedding high-dimensional hypergaphs into a lower-dimensional latent space through GPLVMs as well as a principled approach to selecting inducing vertices for sparse hypergraph GPs. Finally, we demonstrate the enhanced utility of our work through three real-world examples, each of which illustrate distinct aspects of our framework. We envision that the techniques presented in this work are of high utility to both the GP and graph theory communities and will facilitate the blossoming of future research at the intersection of these two areas.

References

Vincent Adam, Stefanos Eleftheriadis, Artem Artemev, Nicolas Durrande, and James Hensman. Doubly sparse variational gaussian processes. In , *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*. PMLR, 2020.

Sameer Agarwal, Jongwoo Lim, Lihi Zelnik-Manor, Pietro Perona, David J. Kriegman, and Serge J. Belongie. Beyond pairwise clustering. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*. IEEE Computer Society, 2005.

Sameer Agarwal, Kristin Branson, and Serge J. Belongie. Higher order learning with graphs. In , *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*. ACM, 2006.

Mauricio A. Alvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for Vector-Valued Functions: A Review. *Foundations and Trends® in Machine Learning*, 4(3):195–266, 2012.

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Comput.*, 10(2):251–276, 1998.

Song Bai, Feihu Zhang, and Philip H. S. Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognit.*, 110:107637, 2021.

Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. Networks beyond pairwise interactions: Structure and dynamics. *Physics Reports*, 874:1–92, 2020.

Austin R. Benson. Three hypergraph eigenvector centralities. *SIAM J. Math. Data Sci.*, 1(2):293–312, 2019.

Cristian Bodnar, Fabrizio Frasca, Yu Guang Wang, Nina Otter, Guido Montúfar, Pietro Liò, and Michael M. Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. *CoRR*, abs/2103.03212, 2021.

Viacheslav Borovitskiy, Iskander Azangulov, Alexander Terenin, Peter Mostowsky, Marc Peter Deisenroth, and Nicolas Durrande. Matern Gaussian Processes on Graphs. *arXiv:2010.15538*, 2020a. arXiv: 2010.15538.

Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Peter Deisenroth. Matérn gaussian processes on riemannian manifolds. In , *Advances in Neural Information Processing Systems 33: Annual Conference on Neural*

Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020b.

Alain Bretto. Hypergraph theory. *An introduction. Mathematical Engineering. Cham: Springer*, 2013.

Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *CoRR*, abs/1611.08097, 2016.

David R. Burt, Carl Edward Rasmussen, and Mark van der Wilk. Convergence of sparse variational inference in gaussian processes regression. *Journal of Machine Learning Research*, 21(131):1–63, 2020.

Fan Chung. The laplacian of a hypergraph. *Expanding graphs (DIMACS series)*, 1993.

Andreas C. Damianou and Neil D. Lawrence. Deep gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 - May 1, 2013*, volume 31 of *JMLR Workshop and Conference Proceedings*. JMLR.org, 2013.

Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. Gpflow: A gaussian process library using tensorflow. *J. Mach. Learn. Res.*, 18:40:1–40:6, 2017.

Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, 2015.

Peter J Diggle, Jonathan A Tawn, and Rana A Moyeed. Model-based geostatistics. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 47(3): 299–350, 1998.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In , *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*. AAAI Press, 1996.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019.

Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, 2016.

Daniel Hernández-Lobato, José Miguel Hernández-Lobato, and Pierre Dupont. Robust multi-class gaussian process classification. In , *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, 2011.

Shenglong Hu and Liqun Qi. The laplacian of a uniform hypergraph. *Journal of Combinatorial Optimization*, 29(2):331–366, 2015.

HyperNetX. HyperNetX: Python package for hypergraph analysis and visualization. https://github.com/pnnl/HyperNetX, since 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In , *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Eric D. Kolaczyk. *Statistical Analysis of Network Data: Methods and Models*. Springer Publishing Company, Incorporated, 1st edition, 2009.

Risi Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In , *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*. Morgan Kaufmann, 2002.

Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In , *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*. MIT Press, 2003.

Sang Hoon Lee, José Manuel Magallanes, and Mason A. Porter. Time-dependent community structure in legislation cosponsorship networks in the Congress of the Republic of Peru. *Journal of Complex Networks*, 5(1):127–144, 2017.

Felix Leibfried, Vincent Dutordoir, S. T. John, and Nicolas Durrande. A tutorial on sparse gaussian processes and variational inference. *CoRR*, abs/2012.13962, 2020.

Finn Lindgren, Håvard Rue, and Johan Lindström. An explicit link between gaussian fields and gaussian markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498, 2011.

Sergey V Lototsky and Boris L Rozovsky. *Stochastic partial differential equations*. Springer, 2017.

Jonas Mockus. *Bayesian Approach to Global Optimization: Theory and Applications*. Springer, Dordrecht, 2012. OCLC: 851374758.

Iain Murray, Ryan Adams, and David MacKay. Elliptical slice sampling. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In , *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. AAAI Press, 2015.

Xuan Vinh Nguyen, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In , *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*. ACM, 2009.

Chris J. Oates, Mark Girolami, and Nicolas Chopin. Control functionals for Monte Carlo integration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):695–718, 2017.

Felix L. Opolka and Pietro Liò. Graph convolutional gaussian processes for link prediction. *CoRR*, abs/2002.04337, 2020.

Manfred Opper and Cédric Archambeau. The Variational Gaussian Approximation Revisited. *Neural Computation*, 21(3):786–792, 2008.

Thomas Pinder, Christopher Nemeth, and David Leslie. Stein Variational Gaussian Processes. *arXiv:2009.12141 [cs, stat]*, 2020. arXiv: 2009.12141.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

Peng Ren, Richard C. Wilson, and Edwin R. Hancock. Spectral embedding of feature hypergraphs. In , *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshop, SSPR & SPR 2008, Orlando, USA, December 4-6, 2008. Proceedings*, volume 5342 of *Lecture Notes in Computer Science*. Springer, 2008.

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, 2007. Association for Computational Linguistics.

Havard Rue and Leonhard Held. *Gaussian Markov random fields: theory and applications.* CRC press, 2005.

Shota Saito, Danilo P. Mandic, and Hideyuki Suzuki. Hypergraph p-laplacian: A differential geometry view. In , *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018.* AAAI Press, 2018.

Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. In , *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, volume 84 of *Proceedings of Machine Learning Research*. PMLR, 2018.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, 2005.

Arno Solin and Simo Särkkä. Hilbert space methods for reduced-rank Gaussian process regression. *Statistics and Computing*, 2019.

Simo Särkkä and Arno Solin. *Applied stochastic differential equations.* Cambridge University Press, 2019. OCLC: 1099853026.

Michalis K. Titsias. Variational learning of inducing variables in sparse gaussian processes. In , *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, volume 5 of *JMLR Proceedings*. JMLR.org, 2009.

Kathryn Turnbull, Simón Lunagómez, Christopher Nemeth, and Edoardo Airoldi. Latent space modelling of hypergraph data, 2019.

Mark van der Wilk, Vincent Dutordoir, S. T. John, Artem Artemev, Vincent Adam, and James Hensman. A framework for interdomain and multioutput gaussian processes. *CoRR*, abs/2003.01115, 2020.

P. Whittle. Stochastic processes in several dimensions. *Bulletin of the International Statistical Institute*, 40:974–994, 1963.

C.K.I. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12): 1342–1351, 1998.

Yin-Cong Zhi, Yin Cheng Ng, and Xiaowen Dong. Gaussian processes on graphs via spectral kernel learning. *CoRR*, abs/2006.07361, 2020.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In , *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*. MIT Press, 2006.

Tinghui Zhou, Hanhuai Shan, Arindam Banerjee, and Guillermo Sapiro. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012*. SIAM / Omnipress, 2012.

A   Full experimental details

A.1   Training configurations

Graph representation   In Section 5.1 and 5.3 our hypergraph kernel is compared against kernels operating on regular graphs. To represent our hypergraph as a graph we use a clique expansion (Appendix A.2). In Section 5.1 we use both a weighted and binary clique expansion and, due to its superior performance, only a weighted expansion in Section 5.3 to compute the diffusion kernel.

Hardware   All experiments are carried out on an Nvidia Quadro GP100 GPU, a 20 core Intel Xeon 2.30GHz CPU and 32GB of RAM.

Implementation   All code is implemented using GPFlow (de G. Matthews et al., 2017) and is made publicly available at
`https://github.com/RedactedForReview`.

Optimisation   For the experiments carried out in Section 5.1 and Section 5.2 we use natural gradients (Amari, 1998) for optimisation of the variational parameters with a learning rate of 0.001 as per Salimbeni et al. (2018). For optimisation of the GP models' hyperparameters in all experiments, we use the Adam optimiser with the recommended learning rate of 0.001 (Kingma and Ba, 2015).

Parameter constraints   For all parameters where positivity is a constraint (i.e. variance), the softplus transformation i.e., $\log(1 + \exp(x))$. Inference is then conducted on the unconstrained parameter, however, we report the re-transformed parameter i.e. the constrained representation.

Parameter initialisation   For the graph kernels we initialise the smoothness, lengthscale and variance parameters to 1.5, 5.0 and 1.0 respectively. For the Gaussian likelihood used in Section 5.2, the variance is initialised to 0.01.

Reported metrics   Given the true test observation y and the predicted value $\hat{y}$, we recall the following definitions: false negative (FN): $\mathbb{1}_{\hat{y}=1\,|\,y=0}$, true negative (TN): $\mathbb{1}_{\hat{y}=0\,|\,y=0}$, false positive (FP): $\mathbb{1}_{\hat{y}=0\,|\,y=1}$, and true positive (TP): $\mathbb{1}_{\hat{y}=1\,|\,y=1}$. Using these definitions, we report the following metrics across the experiments

in for the N heldout observations:

$$\text{Accuracy:} \quad \text{TP} + \text{FP}$$
$$\text{Precision} : \quad \frac{\text{TP}}{\text{TP} + \text{FP}}$$
$$\text{Recall} : \quad \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

In addition to these, we also report the ECE (Naeini et al., 2015)

$$\text{ECE:} \quad \mathbb{E}\left[\Pr\left(\hat{y} = y \,|\, \hat{p} = p\right) - p\right]$$

where $\hat{p}$ is predicted mean.

For the latent space embedding carried out in we use three metrics to quantify the quality of the learned latent space: adjusted mutual information, homogeneity and completeness. *Adjusted mutual information* (Nguyen et al., 2009) is an information theoretic measure that quantifies the degree to which the latent class labels and those generated by the convex hull of each class' latent space coordinates align. Additionally, *homogeneity* measures how homogeneous each cluster's constituent vertices are, whilst *completeness* measures the degree to which all class members were assigned to the correct cluster (Rosenberg and Hirschberg, 2007).

A.2   Clique expansion

Considering a hypergraph $\mathcal{G}$, we can build a regular graph representation using a clique expansion. Let H be the hypergraph's incidence matrix and A_w and A_b be the graph's adjacency matrix for the weighted and binary clique expansion used in , respectively. The $(i, j)^{\text{th}}$ entry of A_w and A_b can be written as

$$\text{Weighted:} \quad [A_w]_{ij} = \sum_{e \in \mathcal{G}} \mathbb{1}(i, j \in \mathcal{G}) \tag{13}$$

$$\text{Binary:} \quad [A_b]_{ij} = \sum_{e \in \mathcal{G}} \mathbb{1}(w_{ij} > 0) \tag{14}$$

Example: Consider the following incidence matrix

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \tag{15}$$

Using the clique expansions in Equation (13) and Equation (14), the corresponding adjacency matrix A_w and A_b can be written as

$$A_w = \begin{bmatrix} 0 & 2 & 2 & 2 & 1 \\ 2 & 0 & 3 & 4 & 2 \\ 2 & 3 & 0 & 3 & 1 \\ 2 & 4 & 3 & 0 & 2 \\ 1 & 2 & 1 & 2 & 0 \end{bmatrix}, \quad A_b = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} \quad (16)$$

A depiction of the matrix representations in Equation (15) and Equation (16) is given in Figure 4.
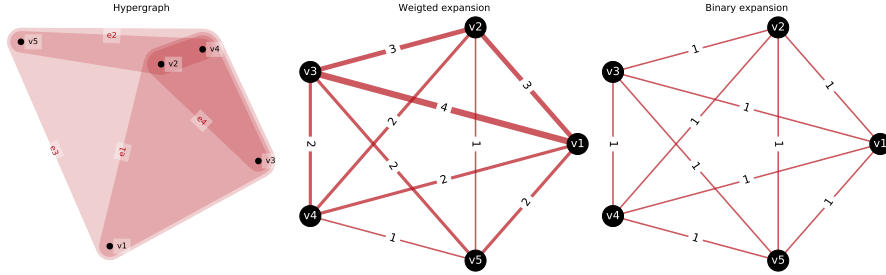

Figure 4: A depiction of the incidence matrix H in Equation (15) and clique expanded adjacency matrices A_w and A_b stated from Equation (16). The width of the edges in the graph are proportional to the edge's weight.

A.3   Diffusion kernel

The corresponding Gram matrix K_D of the diffusion kernel (Kondor and Lafferty, 2002) can be written as

$$K_D = \lim_{n \to \infty} \left(1 - \frac{\beta \Delta}{n} \right)^n \quad (17)$$

where $\beta \in \mathbb{R}$ is the bandwidth parameter and Δ is the (hyper)graph Laplacian matrix. Intuitively, the $(i, j)^{\text{th}}$ entry of K can be seen as the amount of substance that has diffused from v_i to v_j in the original graph. Letting $\beta = 0$ corresponds to zero diffusion and larger values of β will give larger amounts of diffusion. As advised in Zhou et al. (2012), we set $\beta = 0.01$ in Section 5.3.

Computation of the diffusion kernel given in Equation (17) can be achieved through

$$K_D = \exp(-\beta \Delta), \quad (18)$$

as given in Kondor and Lafferty (2002).

23

B Additional experimental results

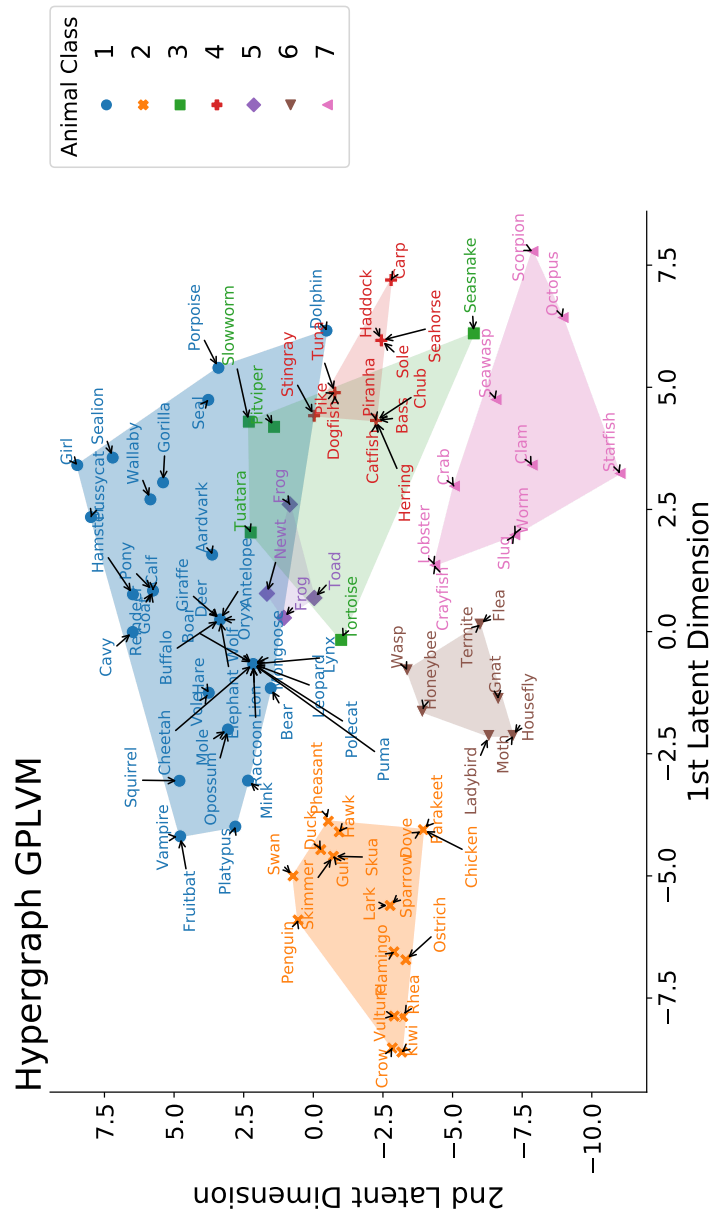## B.1 Latent space embedding

Figure 5: A replication of the hypergraph GPLVM plot given in Figure 3 with every vertex labelled.
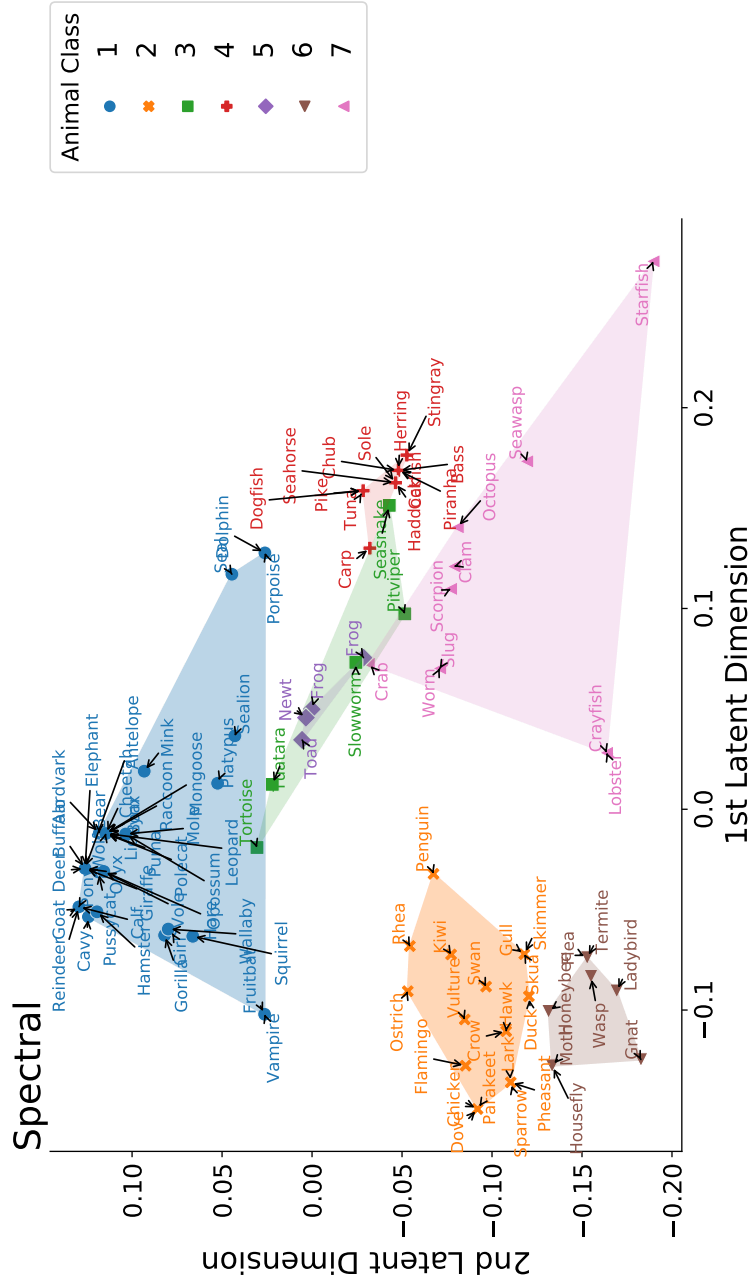
Figure 6: A replication of the spectral plot given in Figure 3 with every vertex labelled.