LaTeX Template for STAT 548 Qualifying Paper Report

Benjamin Bloem-Reddy

October 29, 2023

1 preliminary notes

1.1 abstract and introduction

- use gaussian processes to learn unknown functions "in a manner that permits one to utilize prior information about their properties"
- the idea appears to be that we want a gp for when the input space is an undirected graph
- and the "spde" characterization of matern gps leads to a gp on undirected graphs that is easy to work with
- i think gps are characterized by their mean and (more importantly) covariance functions[1]
- "predicting street congestion with a road network"; this is sort of like predicting a cost or constraint, isn't it?
- "we study GPs whose inputs or outputs are indexed by the vertices of an undirected graph, where each edge between adjacent nodes is assigned a positive weight"
- gaussian markov random fields (gmrf) are relevant here
- it seems that, for graphs with finitely many nodes, the covariance functions can be viewed as "parameterized structured covariance matrices that encode dependence between vertices."
- can we work with infinite graphs, or just finite ones?
- so there have been explorations of kernels for graph gps before, but apparently this attempt to port over a type of matern (which is commonly used for $\mathbb{R}^n$) for graphs is new?
- in "normal" contexts, you can define a matern kernel via spde representation; that appears to be the approach taken for graph gps here
- as an alternative to the spde derivation/version, there is also a fourier feature approach
- graph gp matern kernels converge to euclidean gp matern kernels when the graph becomes more dense?

1.2 gaussian processes

- do i need to worry about the sigma field on the space of functions? I am guessing not for the purposes of this project
- explanation of gp is a bit confusing. $\boldsymbol{x} \in X^n$ is one point $(x_1, \ldots, x_n)$, right? and then $f(\boldsymbol{x}) = f(x_1, \ldots, x_n)$ is a real number, right? do they mean $(f(x_1), \ldots, f(x_n))$?

[1] I think covariance function and kernel are the same thing?

- maybe put anothe rway, i don't understand how the given description jives with the regular notion that a gp is a sequence of rando mvariables such that all finite collections are multivariate-normal distributed
- anyway, we can assume the mean function $\boldsymbol{mu}$ is zero, so forget about that; we focus on the kernel $\boldsymbol{K_{xx}}$
- so the intuitive idea seems to be that we start with some gp (our "prior" gp?), and then given our data, we get a new, updated, "posterior" gp, given the data?
- i guess it's worth noting that both (1) and (2) involve matrix inversion, which I think is computationall expensive
- "mean-square differentiability" seems to be some kind of stochastic process notion of a derivative, which is probably not relevant for this project
- from Wikipedia, it seems that matern kernels are functions of the distance between pairs of points.
- matern gps can be put on the sphere and torus
-

1.2.1 gps on graphs

- i guess if a a matern kernel for a regular gp is a function of the distance between two points, then to create a matern kernel for a graph gp, we need some kind of notion of distance on the graph.
- i guess if we take a step back, then if a gaussian process "is" in some sense a distribution on a space of function, then we can ask: functions from what to what? If $\mathbb{R}^n$ to $\mathbb{R}^n$ is standard, then here we are exploring i guess the possibility that one or both of domain and codomain are weighted graphs?

1.3 matern gps on graphs

- so to define a matern gp on a graph, we need to convert (6) and (7) to some kind of "graph spde?"
- "Note that since a graph is a finite set, such a Gaussian process can be viewed as a multivariate Gaussian whose indices are the graph's nodes"
- so no gps on infinite graphs?
- (8): definition of graph laplacian; so for a finite graph it's just a certain matrix (link)
- don't really get what's going on with the matrix $\Phi(\boldsymbol{\Delta})$. taylor expansions don't always exist, right? and what does it mean to plug a matrix into the taylor expansion of a function from $\mathbb{R}$ to $\mathbb{R}$?
- the mathematical derivations seem to have some serious gaps, at least for my level
-

2 links that might be relevant

- commuting matrices; just thought it seemed interesting; maybe not relevant for this project
- eigendecomp; thinking whether the positive-definiteness of the matrix $\boldsymbol{\Delta}$ is relevant?
- definite matrices; same as above
- orthogonal eigenvectors; related to above

- taylor series; what conditions do we need for Φ to have a taylor expansion?
- matrix exponential; because the matrix exponential appears to be used

3 Publication-style review

3.1 Summary

The paper [Bor+21] introduces a new type of kernel for a Gaussian process on a weighted graph. In particular, the popular Matérn kernel for Gaussian processes on Euclidean space is in some sense converted to a graph-theoretic kernel by replacing the Euclidean Laplace operator with the graph-theoretic Laplacian matrix. Mathematical details of the derivation are not provided in significant detail. As computational efficiency when working with large datasets is a general concern when using Gaussian processes, the authors also discuss computational aspects of the graph Matérn Gaussian process. Sparse graphs generally lead to faster computation, and various approximation techniques are also discussed, both for regression and classification. The authors also provide two comparisons of graph Matérn kernel with other kernels, and the performance is roughly comparable. It is not clear when one kernel might be preferable to another.

3.2 High-level comments on strengths and weaknesses

The main strength of the paper is the introduction of a new (Matérn) kernel for Gaussian processes on graphs. Furthermore, it is at least of theoretical interest that the new kernel is so closely related to the popular Matérn kernel for Gaussian processes on Euclidean space. The main weaknesses of the paper are twofold. First, the graph Matérn kernel does not appear to meaningfully outperform pre-existing alternatives, although this is not a significant weakness. Second, and perhaps slightly more signifcant, the the mathematical detail in [Bor+21] is relatively scarce and it is not obvious that the target audience will be able to reconstruct the mathematical derivations. In relation to other work, [Bor+21] is quite similar at least in inspiration to the authors' previous work [Bor+20], which applies a very similar trick in the replacement of the Euclidean Laplace operator with a related Laplace operator—the Laplace-Beltrami operator in the case of [Bor+20], and the graph Laplacian in the case of [Bor+21].

3.3 Originality

It is not completely obvious precisely what is novel in the paper in question, because there doesn't appear to be any phrase along the lines of "We present a novel. . . " or "We introduce. . . " that would make clear exactly what novel work has been done by the authors. It *appears* that, while Gaussian processes on Euclidean space are well known and Gaussian processes on graphs have been studied, the paper in question is the first to essentially port the Matérn kernel version of a Gaussian process from Eucliean space to the graph context. The work is a combination of prior techniques in the sense that both Gaussian processes on graphs, and Matérn kernels on Euclidean space, have both been studied (separately, apparently). If the work in question is the first to introduce Matérn Gaussian processes on graphs, then it obviously differentiates itself from prior work. Regarding the question of adequate citations, I think it depends to a significant extent on the novelty of the work. If the paper *is* the first to introduce Matérn Gaussian processes on graphs, then by dint of its originality there will not be as as much prior work to cite. If it is not, then the appears to suffer from inadequate citations reflecting this fact, particularly in Sections 2.2 and 3.

3.4 Quality

It is important to keep one's audience in mind when presenting new work. The lead author is a pure mathematician by training and is presenting his work to an audience presumably consisting of statisticians, so there is presumably a large gap in mathematical knowledge beween the author and his audience. It seems very unlikely that many statisticians will have any particular familiarity with, for example, stochastic partial differential equations, heat semigroups, or Riemannian manifolds. Given this gap in mathematical knowledge it appears that the paper in question may be short in detail for some of its mathematical derivations, for example precisely how the standard Matérn kernel for $\mathbb{R}^n$ is converted to to a graph-theoretic version. If statisticians are happy to ignore technical details then the paper might be considered technically sound, especially if we presume that the lead author's mathematical training reduces the chance of his results being erroneous. On the other hand, if the audience is interested in understanding precisely how the graph-theoretic Matérn kernel was derived, then the paper appears short on mathematical justifications. Indeed, the word "proof" does not seem to appear in the paper. The authors appear to honestly present the strengths and weaknesses of their work insofar as they do not claim that Matérn Gaussian processes necessarily represent a major advance, especially since their own empirical results do not appear to show significantly improved performance compared to other kernels.

3.5 Clarity

The writing and presentation generally seem clear, modulo the aforementioned potential issues with a lack of mathematical clarity. The question of whether the paper "adequately informs the reader" of course depends on the reader's level of mathematical sophistication—see the above discussion on this point. Regarding code, the authors have provided code in a GitHub repository.

3.6 Significance

The results appear to be important insofar as they present a new kernel for Gaussian processes on graphs, and in particular it is interesting that a common kernel for Euclidean space has been "ported" over to graphs. The mathematical techniques used appear closely related (at least in spirit) to the techniques employed in the authors' previous paper [Bor+20]

It is unclear whether the mathematical techniques employed in the paper are ripe for further exploitation, at least in part because I don't fully understand then, nor are they thoroughly described. The authors' own experiments do not suggest that graph Matérn Gaussian processes blow away the competition, but with only two comparisons provided, it is too early to judge. At any rate, requiring that a new technique supercede all existing techniques is too high a bar. It is not clear that the authors provide "unique data" or "unique conclusions about existing data", and as described above, it is hard to evaluate their theoretical approach.

3.7 List of questions for the authors

1. Where and how can I learn more about the derivations and convergence results in this paper?

2. Do you have any expectations regarding when the Matérn graph Gaussian process might perform better or worse than alternative kernels, both in terms of accuracy and computational efficiency? (Might the answer to this question depend both on the characteristics of the graph, the dataset itself and the desired type of prediction?)

3. What if anything is impeding the application of Matérn kernels for Gaussian processes on directed and/or infinite graphs?

3.8 Technical/methodological summary

3.8.1 Related literature and broader context

A stochastic process X on parameter space T is said to be *Gaussian* if the random variable $\sum_{i=1}^{n} c_i X_{t_i}$ is Gaussian, for any choice of $n \in \mathbb{N}$, $t_1, \dots, t_n \in T$, and $c_1, \dots, c_n \in \mathbb{R}$ [Kal21]. Gaussian processes have been studied theoretically for at least a century, since the archetypal stochastic process, Brownian motion, is itself Gaussian [Kal21]. However, it appears that only in the last few decades have Gaussian processes become a significant tool in statistics and machine learning [RW06].

By Lemma 14.1 of [Kal21], the distribution of a Gaussian process X is uniquely determined by the functions

$$m_t = \mathrm{E}X_t, \quad r_{s,t} = \mathrm{Cov}(X_s, X_t) \tag{1}$$

for $s, t \in T$. However, among applied practitioners it appears that when using Gaussian processes, much more attention is paid to the covariance function than the mean function [Gar23]. (The precise details regarding the relationship between the regular covariance operation as in (1), and covariance functions more generally, will not be explored in this report.)

The standard definition of a stochastic process is a sequence of random variables $X = \{X_t : t \in T\}$ over a parameter space T, which only requires the existence of some original probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and a (measurable) state space $(S, \mathcal{S})$. This abstract definition permits the possibility that a stochastic process may be defined not only on Euclidean space, but also on other spaces, such as graphs [Gri18]. Gaussian processes in particular are typically employed with Euclidean space as domain and codomain, but there is nothing in principle preventing Gaussian processes being defined on the vertices of a graph $G = (V, E)$, i.e. allowing $\Omega = V$. Indeed, this has been explored, as for example in [KL02]. One of the key challenges in using Gaussian processes on graphs has been finding proper covariance functions, as explained in [KL02], since covariance functions (or "kernels") are required to be symmetric and positive semi-definite.

In the work [Bor+21], a new kernel for Gaussian processes on graphs is explored, in some sense deriving a graph-theoretic version of the popular Matérn kernel for Gaussian processes on Euclidean space. The Euclidean Matérn kernel is a function $C_\nu : \mathbb{R}^d \to \mathbb{R}$ given by

$$C_\nu(d) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{d}{\ell}\right)^{\nu} K_\nu \left(\sqrt{2\nu}\frac{d}{\ell},\right), \tag{2}$$

where d is the Euclidean distance between points $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^n$; $\nu, \ell > 0$; Γ refers to the gamma function and K_ν is the modified Bessel function of the second kind [RW06]. In 3.8.2, we will explore the derivation of the Matérn kernel for Gaussian processes on graphs.

(The paper [Bor+21] also discusses a graph-theoretic version of the squared exponential kernel, which is related to the Matérn kernel. However, due to space limitations, this report will focus on the Matérn kernel.)

3.8.2 Proposed methodology and theoretical properties

There does not appears to be a "proposed methodology" in [Bor+21], insofar as the main contribution is a new kernel for Gaussian processes on graphs—presumably any existing method of using Gaussian processes on graphs can be used with the new kernel. Therefore, this section will focus on explaining whatever theoretical aspects of [Bor+21] that can reasonably be explained, given space limitations.

The paper [Bor+21] refers to the paper [Whi63] in support of a claim that a Gaussian process f on $\mathbb{R}^d$ with Matérn kernel as in (2) satisfies the stochastic partial differential equation

$$\left(\frac{2\nu}{\kappa^2} - \Delta\right)^{\frac{\nu}{2}+\frac{d}{4}} f = \mathcal{W}, \tag{3}$$

where ν and κ are parameters, Δ is the Laplace operator and $\mathcal{W}$ is Gaussian white noise. Unfortunately, the paper [Whi63] cannot be located in either the Simon Fraser University Library, or the University of British Columbia library, so this claim cannot investigated in further detail.

In order to derive a graph-theoretic version of the Matérn kernel, the authors of [Bor+21] focus on the characterization of a Gaussian process as in (3). Although the derivation is not totally clear, a key step in the derivation appears to be replacing the Euclidean Laplace operator Δ with the Laplacian matrix $\boldsymbol{\Delta}$, which is a graph-theoretic analogue of the Euclidean Laplace operator Δ. (Indeed, this appears to be a very similar at least in spirit to the maneuver used by the authors in their previous work [Bor+20], in which they replace the Euclidean Laplace operator in (3) with its Riemannian generalization, the Laplace-Beltrami operator. This substitution leads to the development of a Matérn Gaussian process on Riemannian manifolds in [Bor+20].)

The derivation can be at least partly explained as follows. First assume that the function $\Phi\colon \mathbb{R} \to \mathbb{R}$ as defined in [Bor+21] is analytic, roughly meaning that it is equal to its Taylor series, so we can write

$$\Phi(z) = a_0 + a_1 z + a_2 z^2 + \dots . \tag{4}$$

If we then consider a diagonal matrix

$$\mathbf{D} = \begin{pmatrix} d_1 & 0 & 0 & \dots & 0 \\ 0 & d_2 & 0 & \dots & 0 \\ 0 & 0 & d_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & d_n \end{pmatrix},$$

and we plug this into Φ, then we can write

$$\begin{aligned} \Phi(\mathbf{D}) &= a_0 + a_1\mathbf{D} + a_2\mathbf{D}^2 + \dots \\ &= a_0\mathbf{I} + a_1 \begin{pmatrix} d_1 & 0 & 0 & \dots & 0 \\ 0 & d_2 & 0 & \dots & 0 \\ 0 & 0 & d_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & d_n \end{pmatrix} + a_2 \begin{pmatrix} d_1^2 & 0 & 0 & \dots & 0 \\ 0 & d_2^2 & 0 & \dots & 0 \\ 0 & 0 & d_3^2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & d_n^2 \end{pmatrix} + \dots \\ &= \begin{pmatrix} \Phi(d_1) & 0 & 0 & \dots & 0 \\ 0 & \Phi(d_2) & 0 & \dots & 0 \\ 0 & 0 & \Phi(d_3) & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \Phi(d_n) \end{pmatrix}. \end{aligned}$$

Now, if $\mathbf{O}$ is an orthogonal matrix, then note that

$$\begin{aligned} (\mathbf{ODO}^\intercal)^n &= \underbrace{(\mathbf{ODO}^\intercal)\dots(\mathbf{ODO}^\intercal)}_{n \text{ times}} \\ &= \mathbf{OD}^n\mathbf{O}^\intercal, \end{aligned}$$

so we conclude that

$$\begin{aligned} \Phi(\boldsymbol{\Delta}) &= \Phi(\mathbf{U\Lambda U}^\intercal) \\ &= a_0 + a_1\mathbf{U\Lambda U}^\intercal + a_2(\mathbf{U\Lambda U}^\intercal)^2 + \dots \\ &= \mathbf{U}\Phi(\boldsymbol{\Lambda})\mathbf{U}^\intercal, \end{aligned}$$

which is consistent with (9) in [Bor+21].

The remainder of the argument, namely proceeding from (10) to (11) in [Bor+21], is not totally clear, since it's not obvious where the matrices $\mathbf{U}$ and $\mathbf{U}^\intercal$ have gone. However, it seems possible that their disappearance is related to the invariance of $\boldsymbol{\mathcal{W}} \sim \mathrm{N}(\mathbf{0}, \mathbf{I})$ under orthogonal transformations.

3.8.3 Computational complexity, crucial aspects and potential bottlenecks

Gaussian processes, in and of themselves, are simply stochastic processes as described earlier, so it may only make sense to discuss computational performance in the context of a specific manner of use of Gaussian processes. As such, since [Bor+21] is more focused on introducing a new flavor of Gaussian process, i.e. one with a Matérn kernel on a weighted graph, as opposed to prescribing a specific manner of use, we may begin with generalities and then discuss specific details described in [Bor+21].

Use of Gaussian processes often involves matrix inversion [BDT13; Liu+20], which is a key computational bottleneck in their use. Inversion of an $n \times n$ matrix is generally $O(n^3)$, although faster algorithms, such as Strassen's algorithm, exist [Cor+22]. It is presumably difficult to draw an objective line beyond which datasets become too big for rote use of Gaussian processes, but computational limitations are clearly significant enough that references on Gaussian processes and their use devote attention to efficiency and approximation techniques [RW06; Gar23].

Regarding the specific discussion of computational issues in [Bor+21], a few issues are mentioned, and they are presented below.

First, when the graph G is sparse (having relatively few edges), the graph Laplacian $\boldsymbol{\Delta}$ is a sparse matrix (having relatively few non-zero entries). This leads to sparse precision matrices, which are inverses of covariance matrices [Gar23], and special techniques have been developed to take advantage of matrix sparsity [GL13].

Second, the kernel matrix can be estimated via "truncated eigenvalue expansion", which appears to mean obtaining the ℓ largest eigenvalues of the Matérn kernel, and then proceeding with random features [RR07]. The explanation of this process in [Bor+21] appears to assume familiarity with these techniques. According to [Bor+21], the main drawback of this approach is so-called "variance starvation", which appears to mean that approximation of a Gaussian process can lead to badly underestimated variance as the number of observations increases; see for example Figure 1 in [Wan+18].

Another method concerns modification of the parameter ν in the equation (3), which according to [Bor+21], which roughly speaking controls the smoothness of the sample paths of the Gaussian process. According to [Gar23], the parameter ν is typically given values $k + 1/2$ where $k \in \mathbb{N}$, and higher values lead to smoother sample paths. In contrast, in [Bor+21] it is suggested to set ν to a small whole number value. The succeeding explanation is very brief, but apparently this results in a Gaussian Markov random field, for which training with large amounts of data is relatively straightforward.

For classification, use of Gaussian processes requires some modification because it's less reasonable to assume that likelihoods are Gaussian [RW06]. Again the explanation in [Bor+21] is quite sparse, but the idea appears to be to try to select a representative subset of the data, and calculate an approximation that is as close as possible to that using the entire dataset by minimizing the Kullback-Leibler divergence between two distributions. If the approximation is chosen to be Gaussian, then this amounts to choosing the closest Gaussian process posterior to the true non-Gaussian posterior. (It is not clear what sort of penalty might be paid for this distributional approximation.) According to [Bor+21], there are scalable algorithms for this approach.

3.8.4 Alternative methods that could be applied

In order to discuss "alternative methods that could also be applied to the given problem", we must attempt to define "the problem". Presumably the goal is to do some kind of machine learning on a graph. Below we discuss two cases—Gaussian processes on graphs, and other machine learning techniques on graphs.

Gaussian processes on graphs As [Bor+21] makes clear, Gaussian processes on graphs have been investigated since at least 2002 [KL02], so there are kernels available besides the graph Matérn kernel introduced in [Bor+21]. Based on the comparison provided in [Bor+21], it appears that the graph Matérn kernel is roughly comparable to earlier kernels such as the diffusion kernel, the random walk kernel, and the inverse cosine kernel. However, the comparison in [Bor+21] is not exhaustive, so perhaps a clearer picture of the advantages and disadvantages of different kernels could emerge through further investigation.

Other machine learning techniques on graphs The main methods for machine learning on graphs appear to be graph neural networks, and graph transformers [Fou23; Les23]. A brief persual of Google does not appear to return any comparisons of graph neural networks or graph transformers with Gaussian processes on graphs.

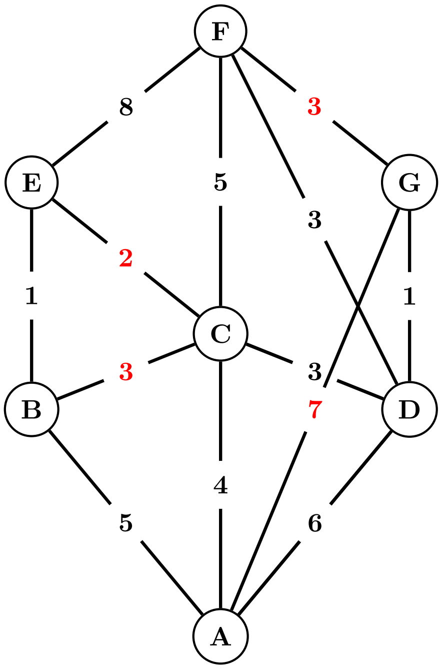

Figure 1: A weighted graph G, representing a simple example of a reinforcement learning state space $\mathcal{S}$ for the mini-project proposal in Section 4. The informal idea is that a true, complete state $s \in \mathcal{S}$ will consist of all weight values and locations in the graph. In contrast, the observation $o \in \mathcal{O}$, $o \subseteq s$ contains only the weights in black. For example, the observation o will contain the weight **6** between nodes **A** and **D**, but not the weight **3** between nodes **B** and **C**. If the weights represent a cost to travel between two nodes, then the agent's task is to find the least costly path to travel from node **A** to node **F**. At each time step, all weights will be drawn from certain probability distributions, and the agent will use a Gaussian process on G to estimate the unknown costs and then select a path from **A** to **F**. Code for graph adapted from TeX StackExchange.

4 Mini-Proposal

The proposed project concerns an application of Gaussian processes on graphs to a partially observable Markov decision process (commonly abbreviated POMDP).

4.1 Area of Opportunity

A brief Google search for "reinforcement learning graph gaussian process" did not return any results that looked comparable.

4.2 Proposed method/approach

The informal idea is that the reinforcement learning agent is managing logistics or route-planning for a concern that must move goods from an origin to a destination on a structure that can be represented by a weighted graph $G = (V, E)$. A weight w between vertices v and v' in V represents the cost of traveling from v to v', and the agent's task is to find the route from origin to destination that minimizes the cost accumulated along the way.

However, at every time step, the costs of traveling between nodes are subject to randomness, and not all costs are observable to the agent. For simplicity, we assume that the agent requires one time step to travel along an edge between nodes. This setup will require the agent to re-calculate its estimates of the costs of routes for the remaining portion of the journey at every time step. This seems to be a reasonable assumption, since it is easy to imagine a delivery van driver having to change alter his planned route because of an (unforeseen, obviously) traffic accident that causes a traffic jam along his planned route.

Even though the Markov decision process is assumed to be only partially observable, suppose the agent has access to a good estimator of its true state. This estimator could take the form of a function $\mathrm{GP}\colon \mathcal{O} \to \mathcal{S}$ that, given an observation o, produces the estimate $\mathrm{GP}(o) = \hat{s}$ of s. Given the estimate $\hat{s}$ of s, the agent may be able to proceed as though its environment is in fact fully observable, and therefore use standard reinforcement learning algorithms that assume full observability. Furthermore, it may be possible train the estimator GP in an online fashion as the agent interacts with its environment, since when the agent acts on information from GP, it may receive one of the true costs it was attempting to estiamte.

As a comparison, the method of estimating the agent's true state and proceeding as though the state is known might be compared with standard techniques for partially observable Markov decision processes.

See Figure 1 for an simple example.

Partially observable Markov decision processes This presentation of partially observable Markov decision processes follows [WO12, Ch. 12].

4.3 Expected technical challenges and bottlenecks

4.3.1 Constructing an appropriate example

One important aspect of this mini-project is finding an appropriate example for testing. An example that is too elaborate may hinder progress, while an example that is too simple may be insufficiently interesting for analysis. Some aspects of example selection requiring judgment are discussed below.

Graph design It is easy to imagine, in general, that a large business may own multiple warehouses across the country, so that in a very large graph, deliveries can in principle be made from any origin (warehouse) node to any other node in the graph. This seems excessively complicated for this mini-project, so instead it seems more reasonable to consider only a simplified graph with one origin node and one destination node, as in Figure 1.

(One simplifying assumption is that an edge between vertices v and v' can be traversed both ways, and furthermore, the costs to travel each way are not separate. This clearly an approximation to reality, since highways can be congested in one direction but not in the other direction. Fixing this assumption would require replacing every edge in a graph as in Figure 1 with two opposite directed edges, with their own cost distributions. However, this is probably an unnecessary complication for this mini-project, especially since the graph Matérn Gaussian process as in [Bor+21] is only specified for undirected graphs.)

Computational cost Related to the above, hopefully an example can be developed that can be simulated on available hardware in a reasonable amount of time. Whether this occurs will only become clear later.

Cost probability distributions The state space $\mathcal{S}$, and hence the observation space $\mathcal{O}$, will naturally have to contain not only the agent's current location in the graph, but also information about the costs on edges between nodes.[2] It is common, though not strictly necessary, in reinforcement learning to assume that the state space $\mathcal{S}$ and action space $\mathcal{A}$ are both finite, for mathematical tractability. To this end, it is necessary to use discrete probability distributions for the costs on each node of the graph.

Furthermore, it is clear that the costs on edges touching the same node cannot reasonably be considered independent. If traffic is heavy between nodes v and v', then that traffic presumably continues to flow elsewhere in the graph, unless one of the nodes is a destination. At this point it is not entirely clear how to enforce this structure. Perhaps costs should be drawn from some sort of discrete approximation to a

[2] In designs with more elaborate graphs, presumably the state space and observation space will also have to include information concerning the delivery origin and destination.

multivariate normal distribution, where covariances between edges decrease as the distance between those edges increases.

4.3.2 Defining the partially observable Markov decision process

Another important aspect of this mini-project is attempting to clearly define the partially observable Markov decision process. To that end, we consider each component in turn below.

State space If the graph contains n edges, then any state $s \in \mathcal{S}$ should presumably be a vector of length $n+1$, containing all current costs and also the agent's current position in the graph. If the cost distribution for the i^{th} edge has $k_i \in \mathbb{N}$ possible values, then the we have the upper bound

$$|\mathcal{S}| \leq |V| \cdot \prod_{i=1}^{n} k_i$$

on the size of the state space.

Action space It seems clear that, given any state $s \in \mathcal{S}$ (and observation $o \in \mathcal{O}$), the actions available to the agent should be movements to nodes connected to the current node. This necessitates a state-dependent action space of the form $\mathcal{A}(s)$, which is somewhat more complicated than is standard. It is clear that

$$|\mathcal{A}(s)| = \deg v,$$

where v is the state component of the vector s.

Observation space

Transition function

Observation function

Cost function The cost function (ordinarily the reward function) should be a function $c\colon \mathcal{S} \times \mathcal{A} \to \mathbb{N}$, giving the cost accrued when taking action a in state s, i.e. the cost on the edge travelled. The cost should be at least partly known to the agent, since some of the costs on the graph edges are known.

4.4 Predicted potential impact

5 Project report

References

[BDT13] A. Banerjee, D. B. Dunson, and S. T. Tokdar. “Efficient Gaussian process regression for large datasets”. In: *Biometrika* 100.1 (2013), pp. 75–89.

[Bor+20] V. Borovitskiy et al. “Matérn Gaussian Processes on Riemannian Manifolds”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 12426–12437. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/92bf5e6240737e0326ea59846a83e076-Paper.pdf.

[Bor+21] V. Borovitskiy et al. “Matérn Gaussian Processes on Graphs”. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by A. Banerjee and K. Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, 13–15 Apr 2021, pp. 2593–2601. URL: https://proceedings.mlr.press/v130/borovitskiy21a.html.

[Cor+22] T. H. Cormen et al. *Introduction to Algorithms*. 4th ed. The MIT Press, 2022.

[Fou23] C. Fourrier. *Introduction to Graph Machine Learning*. Hugging Face. 2023. URL: https://huggingface.co/blog/intro-graphml.

[Gar23] R. Garnett. *Bayesian Optimization*. Cambridge University Press, 2023.

[GL13] G. H. Golub and C. F. V. Loan. *Matrix Computations*. 4th ed. The Johns Hopkins University Press, 2013.

[Gri18] G. Grimmett. *Probability on Graphs: Random Processes on Graphs and Lattices*. 2nd ed. Institute of Mathematical Statistics Textbooks 8. Cambridge University Press, 2018.

[Kal21] O. Kallenberg. *Foundations of Modern Probability*. 3rd ed. Probability Theory and Stochastic Modelling 99. Springer Cham, 2021.

[KL02] R. I. Kondor and J. Lafferty. “Diffusion kernels on graphs and other discrete structures”. In: *Proceedings of the 19th international conference on machine learning*. Vol. 2002. 2002, pp. 315–322.

[Les23] J. Leskovec. *CS224W: Machine Learning with Graphs*. Stanford University. 2023. URL: https://web.stanford.edu/class/cs224w/.

[Liu+20] H. Liu et al. “When Gaussian process meets big data: A review of scalable GPs”. In: *IEEE transactions on neural networks and learning systems* 31.11 (2020), pp. 4405–4423.

[RR07] A. Rahimi and B. Recht. “Random features for large-scale kernel machines”. In: *Advances in neural information processing systems* 20 (2007).

[RW06] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[Wan+18] Z. Wang et al. “Batched large-scale Bayesian optimization in high-dimensional spaces”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2018, pp. 745–754.

[Whi63] P. Whittle. “Stochastic-processes in several dimensions”. In: *Bulletin of the International Statistical Institute* 40.2 (1963), pp. 974–994.

[WO12] M. Wiering and M. Otterlo, eds. *Reinforcement Learning: State of the Art*. Adaptation, Learning and Optimization. Springer Berlin, Heidelberg, 2012.