Approximate Control for Continuous-Time POMDPs

Yannick Eich **Bastian Alt** **Heinz Koeppl**
Department of Electrical Engineering and Information Technology
Technische Universität Darmstadt
{yannick.eich, bastian.alt, heinz.koeppl}@tu-darmstadt.de

Abstract

This work proposes a decision-making framework for partially observable systems in continuous time with discrete state and action spaces. As optimal decision-making becomes intractable for large state spaces we employ approximation methods for the filtering and the control problem that scale well with an increasing number of states. Specifically, we approximate the high-dimensional filtering distribution by projecting it onto a parametric family of distributions, and integrate it into a control heuristic based on the fully observable system to obtain a scalable policy. We demonstrate the effectiveness of our approach on several partially observed systems, including queueing systems and chemical reaction networks.

1 INTRODUCTION

Partial observability in dynamical systems is a ubiquitous problem for many applications. This includes settings such as robotics (Lauri et al., 2022), communication systems and signal processing (Proakis and Salehi, 2008; Kay, 1993), or biology (Wilkinson, 2018). In partially observable settings only noisy data from a latent time-dependent process is available. A principled way to deal with the resulting inference problem is Bayesian filtering (Bain and Crisan, 2009; Särkkä, 2013). The framework of Bayesian filtering can be exploited to infer in an online manner the latent state of the system given the historical information available. The information of the latent state is then encoded in the filtering posterior distribution, the belief state. This stochastic filtering approach is especially appealing for the control of such partially observed dynamical systems. This includes among others, e.g., control problems with noisy sensor measurements, such as grasping and navigation in robotics (Kurniawati et al., 2008) or cognitive medium access control (Zhao et al., 2005) for communication systems. For finding decision strategies, which use the available observational data to control the system at hand, a solid framework can be found in the area of optimal control (Stengel, 1994). The classical setting for partially observable problems in optimal control theory is historically a continuous-time setting with a real-valued stochastically evolving latent state (Bensoussan, 1992). This includes the well-known linear quadratic Gaussian (LQG) control problem, dating back to the work of Wonham (1968). Contrary to the continuous modeling approach, a discrete-time and discrete-state setting has historically been discussed in the area of operations research (Åström, 1965), and has ever since found popularity as partially observable Markov decision process (POMDP) within the machine learning community (Kaelbling et al., 1998).

Over the years numerous algorithms for solving partially observable control problems, especially for discrete-time settings have been proposed. For example, the work of Zhou et al. (2010), which is closely related to ours, uses projection filtering to represent the belief to perform optimal control for continuous state and action spaces. Similarly, other approximate filtering methods, such as particle filtering (Thrun, 1999), have been exploited in a discrete-time context. For the discrete-time setting with discrete spaces, Monte Carlo tree search methods have shown substantial success, such as DESPOT (Somani et al., 2013) and POMCP (Silver and Veness, 2010). Also, more recently methods solving POMDPs in a discrete-time regime using modern deep-learning techniques such as the algorithms of Igl et al. (2018) and Singh et al. (2021), have shown to find good control strategies. However, many problems can neither be modeled by a discrete-time approach nor using continuous-time and space strategies such as the LQG setup. Consider for example the control of

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

a queueing network within a communication system (Bolch et al., 2006). Here the state space is discrete as the system is described by the discrete number of packets in each queue. However, as an instance of a discrete event system, it can not be directly described by a discrete-time modeling approach as the packets arrive and are serviced at non-equidistant time points. Another example is the modeling of molecules in a biological system. In the context of systems biology (Wilkinson, 2018) the dynamical system is described by the discrete number of each of the molecules in the system. The molecule count is dynamically changed due to reactions occurring in the system, which as a physical system evolves continuously in time. When the control of such systems is of interest, it poses a problem for both classical setups and the derived algorithms. Even though there exist strategies to transform these continuous-time discrete-state space problems to one of both setups they are often not sensible and subject to large modeling errors. For example, Langevin dynamics or a deterministic fluid limit are methods used to approximate discrete states by continuous ones, see, e.g., Gardiner (2009) and Ethier and Kurtz (2009). Although these approximations have been used, e.g., in the context of heavy traffic in queueing networks (Kushner, 2001), they fail when the latent state is inherently discrete. This is the case, when the ordinal number representing the state is small or in the extreme case when on-off switching behavior occurs in the system, as the state can not sufficiently be described by a continuous value. Naively discretizing time also has its downside, as many algorithms derived under a discrete-time assumptions are often not robust to time discretization (Tallec et al., 2019). Contrary to that, continuous-time models can numerically be solved using elaborate numerical differential equation solvers, that automatically adapt, e.g., the step size to the specific problem.

The control for fully observed continuous-time discrete-state space Markov decision processes (MDPs) has been discussed in the context of semi-Markov decision processes (SMDPs), e.g., for the control of queueing networks (Bertsekas, 2012a,b), see also Du et al. (2020) and the references therein. However, partial observability in this setting has received little attention by the machine learning community. Recently, Alt et al. (2020) discussed the theory to model partially observed continuous-time discrete-state space problems. However, a substantial scalability issue of their approach is that it has to solve a high-dimensional partial differential equation (PDE) in belief space. Additionally, the presented approach leverages exact filtering within the control problem. These two algorithmic choices make the presented method doubly intractable, as both the exact filter and the control problem in belief space suffer from the curse of dimensionality.

Hence, our contributions are: We provide a new scalable algorithm for the solution of continuous-time discrete-state space POMDPs. Our new method can be divided into two parts: First, we approximate the filtering distribution by a parametric distribution using the method of entropic matching (Bronstein and Koeppl, 2018b). Here we give closed-form solutions for the evolution of the parameters in some interesting problem settings, such as queueing networks and chemical reaction networks (CRNs). Second, to get a scalable control law we adapt a heuristic used historically within discrete-time partially observable systems to the continuous-time POMDP framework described in Alt et al. (2020). This enables us to consider control problems with an unbounded number of states, well beyond the setup of Alt et al. (2020). An implementation of our proposed method is publicly available[1].

2 MODEL

We consider the problem of optimal decision-making under partial observability in continuous time $t \in \mathbb{R}_{\geq 0}$. For this, we exploit a continuous-time POMDP model (Alt et al., 2020), where the latent state trajectory $X_{[0,\infty)} := \{X(t) \in \mathcal{X} \mid t \in \mathbb{R}_{\geq 0}\}$ is modeled as a controlled continuous-time Markov chain (CTMC) (Norris, 1998) on a countable state space $\mathcal{X} \subseteq \mathbb{N}^n$. The rate function for a state $x \neq x'$ of the CTMC is given as

$$\begin{aligned}&\Lambda(x, x', u, t)\\ &\quad := \lim_{h \to 0} h^{-1} \operatorname{P}(X(t+h) = x' \mid X(t) = x, u(t) = u),\end{aligned}$$

where we assume time homogeneity, i.e., $\Lambda(x, x', u, t) \equiv \Lambda(x, x', u), \forall t$. In the above equation the state trajectory $X_{[0,\infty)}$ can be controlled by a control trajectory $u_{[0,\infty)} := \{u(t) \in \mathcal{U} \mid t \in \mathbb{R}_{\geq 0}\}$. Throughout this work, we assume a finite action space setting, i.e., $\mathcal{U} = \{1, 2, \ldots, m\}$, where m denotes the number of actions. We assume that $X(t)$ can not be directly observed but only a partial observation $Y(t) \in \mathcal{Y}$ is available. The goal of an optimal control sequence is then to maximize a reward function $R : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ over a time horizon, i.e.,

$$\underset{u_{[0,\infty)}}{\text{maximize}} \quad \mathsf{E}\left[\int_0^\infty e^{-\frac{t}{\tau}} R(X(t), u(t))\, \mathrm{d}t\right],$$

where we assume an infinite horizon setting with inverse exponential discount variable $\tau \in \mathbb{R}_{\geq 0}$. Note that, in the POMDP setting the control $u(t)$ at time point t can only depend on the observation and control history $y_{[0,t]} := \{y(s) \mid s \in [0,t]\}$ and $u_{[0,t)} := \{u(s) \mid s \in [0,t)\}$, respectively, i.e., $u_{[0,\infty)}$ has to be an admissible control trajectory. Mathematically, we let $u_{[0,\infty)} \in$

[1]`https://github.com/yannickeich/ApproxPOMDPs`

$\mathcal{D}_{\mathcal{F}^u_y}([0,\infty),\mathcal{U})$, where $\mathcal{D}_{\mathcal{F}^u_y}([0,\infty),\mathcal{U})$ is the space of càdlàg functions on $[0,\infty)$ taking values in $\mathcal{U}$ that are adapted to the family of control dependent sigma-algebras $\mathcal{F}^u_y(t) = \sigma(Y(s) \mid s \leq t)$ of the observation process, i.e., indicating that $u(t)$ is $\mathcal{F}^u_y(t)$-measurable (Bensoussan, 1992).

It is well known that the partial observable control problem can be cast as a problem of optimal control in belief space (Bertsekas, 2012a,b). The belief at time point t is the Bayesian filtering distribution

$$\pi_t(x) = \mathrm{P}(X(t) = x \mid y_{[0,t]}, u_{[0,t)}).$$

This yields the optimal feedback policy $u(t) = \mu^*(\pi_t)$.

In the next sections, we explain how to find both the optimal feedback policy, as well as how to compute the exact filtering distribution. However, both problems are computationally intractable. Hence, we present two approximations for the solution of (i) the optimal filtering problem and (ii) the optimal control problem.

3 CONTINUOUS-TIME PROBABILISTIC INFERENCE FOR POMDPS

The state of the underlying belief MDP is characterized by the filtering distribution $\pi_t(x)$. For computing the filtering distribution, we first consider the prior distribution for the latent process $X_{[0,\infty)}$. We can characterize it by its time-point-wise marginal distribution $p_t(x) \coloneqq \mathrm{P}(X(t) = x \mid u_{[0,t)})$. For a controlled CTMC $X_{[0,\infty)}$, with action trajectory $u_{[0,\infty)}$ and rate function Λ, the time evolution of the prior is given by the differential form of the forward Chapman-Kolmogorov equation (Ethier and Kurtz, 2009), the *master equation* (Gardiner, 2009),

$$\frac{\mathrm{d}}{\mathrm{d}t} p_t(x) = [\mathcal{L}_{u(t)} p_t](x) \tag{1}$$

with initial distribution $p_0(x)$ at time point $t = 0$. The evolution operator $\mathcal{L}_u$ of the Markov chain is given as

$$\begin{aligned}&[\mathcal{L}_u \phi](x)\\ &\quad\coloneqq \sum_{x' \neq x} \left\{ \Lambda(x', x, u)\phi(x') - \Lambda(x, x', u)\phi(x) \right\}\end{aligned}$$

for an arbitrary test function ϕ. Next, we have to specify the generative model for the observation process $\{Y(t) \mid t \in \mathbb{R}_{\geq 0}\}$. In this paper, we will consider two observation models, which are (i) a discrete-time noisy measurement model and (ii) a sub-system measurement model.

Noisy Measurements. A natural observation model to consider is a discrete-time noisy measurement model, which we denote as D. For this we assume that the observations $Y(t)$ are given at discrete time instances $\{t_i\}$, i.e., $\{Y_i \coloneqq Y(t_i) \mid i \in \mathbb{N}\}$. Further, we assume that the state is not directly observed but only a noisy measurement is available as

$$D: \quad Y_i \mid \{X(t_i) = x, u(t_i^-) = u\} \sim p(y_i \mid x, u), \tag{2}$$

where throughout this paper we denote by $\phi(t_i^-) \coloneqq \lim_{t \nearrow t_i} \phi(t)$ the limit from the left of an arbitrary function ϕ, respectively. Note that this observation model gives rise to a continuous-discrete filtering problem, i.e., filtering for latent states evolving in continuous-time and discrete-time observations, for more see, e.g., Maybeck (1982), and Särkkä and Solin (2019).

Sub-System Measurements. Additionally we consider a sub-system measurement model, which we denote by C. For this we assume that we can only observe some components of the n-dimensional random vector $X(t)$. Consider without loss of generality that $X(t) = [\hat{X}^\top(t), \bar{X}^\top(t)]^\top$, and that the components $\hat{X}(t)$ and $\bar{X}(t)$ are unobserved and observed, respectively. Hence, as for the observations $\{Y(t) \mid t \in \mathbb{R}_{\geq 0}\}$ we have the noise-free measurement model

$$C: \quad Y(t) = \bar{X}(t). \tag{3}$$

As the process $\{X(t)\}$ evolves on a countable space, this implies that at the discrete-time instances $\{t_i\}$ we observe a jump from a state $\bar{X}(t_i^-)$ to state $\bar{X}(t_i)$, i.e., we have a discrete-time noise-free observation as $Y_i \coloneqq Y(t_i) = \bar{X}(t_i)$. Note that this observation model (Bronstein and Koeppl, 2018a) can be seen as a generalization of a Poisson process observation model (Elliott and Malcolm, 2005).

3.1 Exact Inference

When considering the task of computing the filtering distribution, the sought after posterior distribution can be calculated by Bayes' rule. This yields for the noisy measurement model D in Eq. (2) a differential equation for the filter between observations as

$$D: \quad \frac{\mathrm{d}}{\mathrm{d}t} \pi_t(x) = [\mathcal{L}_{u(t)} \pi_t](x). \tag{4}$$

At observation time points, Bayes' rule is computed as

$$D: \quad \pi_{t_i}(x) = Z_i^{-1} p(y_i \mid x, u(t_i^-)) \pi_{t_i^-}(x), \tag{5}$$

where $Z_i = \sum_x p(y_i \mid x, u(t_i^-)) \pi_{t_i^-}(x)$ is a normalization constant, for more see Huang et al. (2016). The resulting filtering equation is the usual result in continuous-discrete filtering, as the *prediction step* in Eq. (4) between the observations is given by the time evolution of the prior distribution, see Eq. (1), and the *update step* in Eq. (5) carries out Bayes' rule for the observation y_i.

For the sub-system measurement model C in Eq. (3), we find analog equations between jumps of the observation process, i.e., when the observation process is constant $y(t) = y(t^-)$,

$$C: \quad \begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\pi_t(x) &= \mathbb{1}_{y(t)}(x)[\mathcal{L}_{u(t)}\pi_t](x) \\ &\quad - \pi_t(x)\sum_{x'} \mathbb{1}_{y(t)}(x')[\mathcal{L}_{u(t)}\pi_t](x'), \end{aligned} \tag{6}$$

where $x = [\hat{x}^\top, \bar{x}^\top]^\top$ and $\mathbb{1}_{y(t)}(x) \coloneqq \mathbb{1}(y(t) = \bar{x})$. When the observation process jumps $y_i = y(t_i) \neq y(t_i^-)$, we have

$$C: \quad \pi_{t_i}(x) = Z_i^{-1}\,\mathbb{1}_{y_i}(x)[\mathcal{L}_{u(t)}\pi_{t_i^-}](x), \tag{7}$$

with the normalization constant $Z_i = \sum_x \mathbb{1}_{y_i}(x)[\mathcal{L}_{u(t)}\pi_{t_i^-}](x)$, for more see Bronstein and Koeppl (2018a).

However, all these equations suffer from the curse of dimensionality as the numerical solution of the system of ordinary differential equations (ODEs) in Eqs. (4) and (6) scales in $|\mathcal{X}|^2$, which in the case $\mathcal{X} = \mathbb{N}^n$ is even infinite. Similarly, computing the update steps in Eqs. (5) and (7) requires computing the normalization constant Z_i, which is intractable. Hence, we propose next an approximate inference method for the solution to the intractable exact filtering problem.

3.2 Approximate Inference

In order to overcome the challenge of computing the exact intractable filtering distribution, we propose to approximate it by a parametric distribution using an assumed density filtering method (Maybeck, 1982). For this we use a deterministic approximation method, entropic matching (Ramalho et al., 2013; Bronstein and Koeppl, 2018b), which can be seen as a continuous-time extension of a special expectation propagation method (Minka, 2005). As a result, we obtain low-dimensional time evolution equations for the parameters of the distribution.

We start by assuming a parametric form for the filtering distribution at time t as $\pi_t(x) \approx q_{\theta(t)}(x)$, with parameters $\theta(t) \in \Theta \subseteq \mathbb{R}^{d_\theta}$. We then consider the evolution of the distribution $q_{\theta(t)}(x)$ over a small time step h in absence of new measurements. For the observation model D, the distribution follows the prediction step in Eq. (4), which yields

$$\tilde{\pi}_{t+h}(x) \coloneqq q_{\theta(t)}(x) + h[\mathcal{L}_{u(t)} q_{\theta(t)}](x) + o(h).$$

It is worth noting that $\tilde{\pi}_{t+h}(x)$ generally does not match the structure of the parametric family used for approximation. To obtain a distribution within the parametric family, its optimal parameters after a small time step h can be computed by minimizing the reverse Kullback-Leibler (KL) divergence

$$\theta(t+h) = \operatorname*{arg\,min}_{\theta'} \mathsf{KL}(\tilde{\pi}_{t+h} \parallel q_{\theta'}).$$

The entropic matching method approximates the change of the filtering distribution in an infinitesimal time step such that the approximated distribution stays in the space of the parametric distributions. By computing the continuous-time limit $\lim_{h\to 0} h^{-1}\{\theta(t+h) - \theta(t)\}$ an ODE for the parameters can be found (Bronstein and Koeppl, 2018b) as

$$D: \quad \begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) &= F(\theta(t))^{-1} \\ &\quad \cdot \mathsf{E}_q\left[\mathcal{L}^\dagger_{u(t)} \nabla_{\theta(t)} \log q_{\theta(t)}(X(t))\right], \end{aligned} \tag{8}$$

where $F(\theta)$ is the Fisher information matrix of the parametric distribution, i.e., $F(\theta) = \mathsf{E}_q\left[\nabla_\theta \log q_\theta(X) \nabla_\theta^\top \log q_\theta(X)\right]$. The score $\nabla_{\theta(t)} \log q_{\theta(t)}(x)$ evolves according to the operator $\mathcal{L}_u^\dagger$

$$[\mathcal{L}_u^\dagger \psi](x) \coloneqq \sum_{x' \neq x} \Lambda(x, x', u)\left\{\psi(x') - \psi(x)\right\},$$

for an arbitrary test function ψ, where $\mathcal{L}_u^\dagger$ is adjoint to the evolution operator $\mathcal{L}_u$, w.r.t. the inner product space $\langle \phi, \psi \rangle = \sum_x \phi(x)\psi(x)$. At the observation jump time points, we have a discontinuity in the filtering distribution as in the update step in Eq. (5). Here we find the optimal parameters accordingly as

$$\theta(t_i) = \operatorname*{arg\,min}_{\theta'} \mathsf{KL}(\hat{\pi}_{t_i} \parallel q_{\theta'}), \tag{9}$$

where $\hat{\pi}_{t_i}$ is the posterior distribution that follows from Bayes' rule

$$D: \quad \hat{\pi}_{t_i}(x) \coloneqq Z_i^{-1} p(y_i \mid x, u(t_i^-))\, q_{\theta(t_i^-)}(x),$$

with $Z_i = \sum_x p(y_i \mid x, u(t_i^-))\, q_{\theta(t_i^-)}(x)$. For the observation model C, we have for the time evolution

$$C: \quad \begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) &= F(\theta(t))^{-1} \\ &\quad \cdot \mathsf{E}_q\left[\mathcal{L}^\dagger_{u(t)}\left\{\mathbb{1}_{y(t)} \cdot \nabla_{\theta(t)} \log q_{\theta(t)}\right\}(X(t))\right]. \end{aligned} \tag{10}$$

For the reset condition, we have the same objective as in Eq. (9), however, the reset is computed by minimizing the reverse KL divergence w.r.t.

$$C: \quad \hat{\pi}_{t_i}(x) \coloneqq Z_i^{-1}\,\mathbb{1}_{y_i}(x)[\mathcal{L}_{u(t)} q_{\theta(t_i^-)}](x),$$

with $Z_i = \sum_x \mathbb{1}_{y_i}(x)[\mathcal{L}_{u(t)} q_{\theta(t_i^-)}](x)$, for more see Bronstein and Koeppl (2018a). For completeness we give the derivation of Eqs. (8) and (10) in Appendix 1.1. We want to emphasize that these equations can also be derived from a geometrical approach, which is then known as projection filtering (Brigo et al., 1999).

The resulting equations for the time evolution in Eqs. (8) and (10) require only the solution of an ODE in the parameter space Θ. Therefore, the computational complexity is only d_θ^2 instead of $|\mathcal{X}|^2$. Further, when assuming an exponential family parameterization, the optimal variational distribution at the discrete observation time points in Eq. (9) reduces to the problem of moment matching (Bishop, 2006).

4 CONTINUOUS-TIME CONTROL FOR POMDPS

Next, we describe how an optimal control trajectory $u^*_{[0,\infty)}$ can be found. First, we consider the problem of exact decision-making, which is however, intractable. Therefore, we present an approximate control method.

4.1 Exact Control

We can describe the POMDP by a belief MDP using the filter distribution as the continuous state. For the sake of presentation we assume for now, that the state space $\mathcal{X}$ is finite, therefore, we can represent the filtering distribution at time point t as a vector $\pi_t \in \Delta^{|\mathcal{X}|}$, with components $\{\pi_t(x) \mid x \in \mathcal{X}\}$, where $\Delta^{|\mathcal{X}|}$ is the $|\mathcal{X}|$ dimensional probability simplex. This allows us to use stochastic optimal control theory for continuous-valued states. We define the value function for a belief $\pi \in \Delta^{|\mathcal{X}|}$ as the expected cumulative reward under the optimal control, i.e.,

$$V(\pi) \\ := \max_{u_{[t,\infty)}} \mathsf{E}\left[\int_t^\infty \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s))\, \mathrm{d}s \,\middle|\, \pi_t = \pi\right],$$

where we use a normalization by $\frac{1}{\tau}$. Exploiting the principle of optimality, a Bellman equation can then be found in the form of a Hamilton-Jacobi-Bellman (HJB) equation (Alt et al., 2020) as

$$V(\pi) = \max_{u\in\mathcal{U}} \mathsf{E}\Bigg[R(X,u) + \tau \frac{\partial V(\pi)}{\partial \pi} f(\pi, u) \\ + \tau\lambda(X,u)(V(\pi + h(\pi,u)) - V(\pi)) \,\Bigg|\, \pi\Bigg], \tag{11}$$

where the functions f, h and λ are defined by the filter dynamics, for more see the work of Alt et al. (2020). An optimal policy can be retrieved by finding the maximizer of the r.h.s. of Eq. (11).

Hence, finding the optimal policy requires solving for the value function, which is generally hard, as it requires solving a $|\mathcal{X}|$-dimensional PDE. Therefore, even methods based on learning with function approximation, such as the one used in the work of Alt et al. (2020) scale very poorly with the state-space size. Additionally, even when using approximate filter dynamics, as the ones previously discussed in Section 3.2, the r.h.s. of the HJB equation in Eq. (11) requires solving expectations over the state, as well as the observation space. This makes numerical solution methods, including learning-based PDE solution methods, intractable for all but very small state and observation space problems. For this reason, we present next an approximate control method, which does not require the solution of a high-dimensional PDE.

4.2 Approximate Control

Instead of computing the optimal control based on the dynamics of the filtering distribution we approximate it by separating the filtering and the control problem. First, we compute the value function for the underlying MDP. We then combine it with the filtering distribution to approximate the value function of the POMDP to retrieve a policy. In the discrete-time literature, this is known as the QMDP method (Littman et al., 1995), which has also recently found success using function approximation (Karkus et al., 2017). Although the method assumes full observability for the planning and therefore does not consider the information gathering effect of actions, it leads to good results in many examples (Littman et al., 1995).

Similar to the belief MDP we define the value function of the underlying MDP for a state $x \in \mathcal{X}$ as the expected cumulative reward under the optimal control, i.e.,

$$V(x) \\ := \max_{u_{[t,\infty)}} \mathsf{E}\left[\int_t^\infty \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s))\, \mathrm{d}s \,\middle|\, X(t) = x\right],$$

where for the underlying MDP we assume that the admissible control $u(t)$ can depend on the state $X(t)$. By the principle of optimality the Bellman equation for continuous time and discrete state space (Bertsekas, 2012a,b) is

$$V(x) = \max_{u\in\mathcal{U}} R(x,u) + \tau \sum_{x'\neq x} \Lambda(x,x',u)(V(x') - V(x)). \tag{12}$$

The optimal policy for the MDP maximizes the r.h.s. of Eq. (12), which we define as state-action value function

$$Q(x,u) := R(x,u) + \tau \sum_{x'\neq x} \Lambda(x,x',u)(V(x') - V(x)).$$

The definition of the state-action value function above can be reformulated as a contraction mapping, as

$$Q(x,u) = \frac{R(x,u)}{1+\tau\sum_{x'\neq x}\Lambda(x,x',u)} + \tau\sum_{x'\neq x}\frac{\Lambda(x,x',u)}{1+\tau\sum_{x'\neq x}\Lambda(x,x',u)}\max_{u'\in\mathcal{U}} Q(x',u').$$

The derivation is provided in Appendix 2.1. When the state space is finite, above equation can be solved using fixed point iteration, similar to tabular dynamic programming methods like the value iteration algorithm in the discrete-time setting (Sutton and Barto, 2018). For large or even infinite state spaces the fixed point iteration becomes intractable and we use value function approximation methods (Bradtke and Duff, 1994). For implementation details see Appendix 2.2.

An approximate optimal policy for the POMDP is then found as the maximizer of the expected state-action value function i.e.,

$$u(t) = \arg\max_{u'\in\mathcal{U}} \mathsf{E}_q[Q(X(t),u')], \qquad (13)$$

where the expectation is w.r.t. the approximate filter distribution. For large or even infinite state spaces, where the expectation is intractable, we approximate it by Monte Carlo sampling to obtain a scalable policy.

5 EXPERIMENTS

To evaluate the efficacy of our method for partially observed systems we test it on three continuous-time discrete-state space control problems. We evaluate (i) a controlled queueing network, (ii) a controllable predator-prey system in form of a Lotka-Volterra (LV) model, and (iii) a controlled closed-loop four species CRN.

5.1 Queueing Network

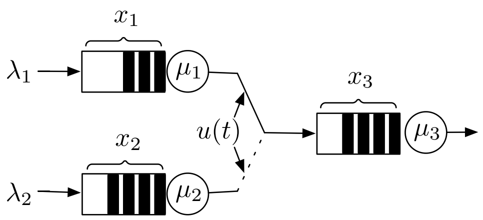

Figure 1: A schematic description of the considered queueing problem. The decision-maker decides which queue outputs its packets to the third queue.

First, we consider a queueing problem consisting of $n = 3$ queues with fixed buffer size $N = 1000$, i.e., $\mathcal{X} = \{0,1,\dots,N\}^n$. The queues are connected as displayed in Fig. 1. Packets arrive with constant rates λ_1 and λ_2 in the first and the second queue, respectively.

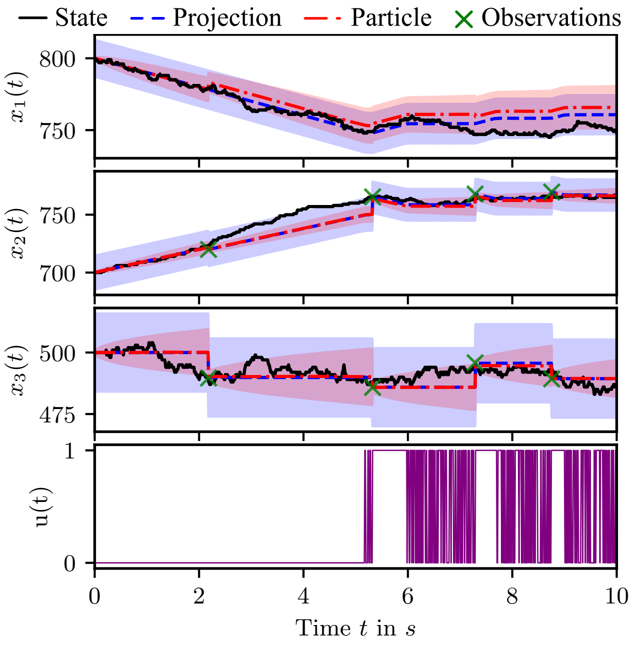

Figure 2: A sample trajectory of the queueing problem using a policy computed by the QMDP method. The upper plots compare the projection filter to a particle filter by indicating their mean and variance. The lower plot describes the actions over time.

The action $u(t)$ decides which of the two queues (with service rates μ_1 or μ_2) preprocesses packets before sending them to the final queue. If $u(t) = 0$, queue 1 is servicing queue 3; if $u(t) = 1$, queue 2 is servicing queue 3. The preprocessed packets are then being handled with service rate μ_3. As an observation model, we use Gaussian discrete-time measurements of queue 2 and 3. The reward model is designed to favor empty queues. The parameters, reward function, observation model and additional information to all experiments can be found in Appendix 3.

We use entropic matching to approximate the $(N+1)^n \approx 10^9$-dimensional exact filtering distribution by a binomial distribution for each queue $q_\theta(x) = \prod_{i=1}^n \mathrm{Bin}(x_i \mid N, \theta_i)$, with success probabilities $\{\theta_i\}_{i=1}^n$ and the total number of trials N is fixed to the maximal buffer size. Closed-form solutions for the drift of the binomial parameters for general queuing systems with finite buffer sizes are derived in Appendix 1.2. The impact on the parameters by the measurements can be computed by Eq. (9), which we approximate by moment matching right before and after the measurement to get closed-form updates, see Appendix 1.2.

The quality of the approximate filtering distribution plays a crucial role in the performance of our control method. To validate the effectiveness of our approach, we compare our projection filter against a bootstrap particle filter. Notably, the particle filter, configured with a sufficiently large number of samples $N_s = 10000$, serves as a robust benchmark or "ground truth" for our comparison. Figure 2 shows the mean and variance of both filters for a sample trajectory. The comparison highlights the strengths and the limitations of the projection filter. While it successfully captures the evolution of the mean induced by the prior dynamic, it struggles to capture the dynamics of the variance, due to the limited expressiveness of the parametric distribution. A second approximation error is noticeable at the observation times. As the distributions for each queue are modeled to be independent, there is no update in the first queue, when the other queues are being observed. However, despite these limitations, the projection filter leads to an adequate approximation.

For all experiments, the policy is chosen according to our QMDP method. Therefore we learned the state action value function of the underlying MDP and approximate the expectation in Eq. (13) using $k = 20$ Monte Carlo samples. The resulting policy effectively maximizes the cumulative reward by activating the service rates of the queue where the filter expects more packets. When the beliefs of the first and second queue are equal, the policy frequently jumps between both actions, attempting to maintain the beliefs at the same level until a new observation updates them.

5.2 Predator-Prey System

Next, we consider the continuous-time discrete-state LV model of Wilkinson (2018). The LV problem consists of the following reactions, which are represented using the notation of CRNs (Wilkinson, 2018) as

$$\mathsf{X}_1 \xrightarrow{c_1} 2\mathsf{X}_1, \quad \mathsf{X}_1 + \mathsf{X}_2 \xrightarrow{c_2} 2\mathsf{X}_2, \qquad \mathsf{X}_2 \xrightarrow{c_3(u(t))} \emptyset, \tag{14}$$

where X_1 is the prey species and X_2 is the predator species. Hence, the unbounded state space of the system is $\mathcal{X} = \mathbb{N}_0^n$, with $n = 2$. As an option to control the system, the decision maker can influence rate c_3 with $u(t) \in \mathcal{U} = \{0, 1\}$, where $c_3(u(t) = 1) = 2c_3(u(t) = 0)$. For the observation model, we use noisy discrete-time measurements of both states. The reward model returns the negative Euclidean distance to a fixed target state $x^* = [100, 100]^\top$. As an approximation to the infinite-dimensional filtering distribution we use a product Poisson distribution as $q_\theta(x) = \prod_{i=1}^n \mathrm{Pois}(x_i \mid \theta_i)$. Bronstein and Koeppl (2018b) showed that using the entropic matching method with a product Poisson approximation leads to closed-form solutions for the drift of the parameters for general CRNs. In Appendix 1.3 we show generalize the derivation to include actions and provide details on CRNs.

Using the QMDP method with the approximate filtering distribution we can compute the advantage function $A(\theta, u) \coloneqq \mathsf{E}_q[Q(X, u) - V(X)]$, visualized in Fig. 3. From this we can retrieve the policy as the actions that maximize the advantage function.

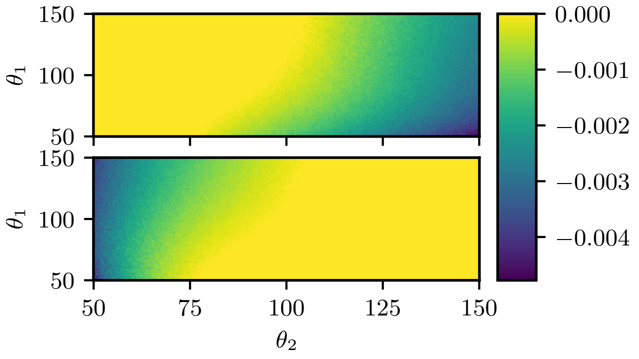

Figure 3: Advantage function for the LV problem. The upper and the lower plot show the advantage function over a section of the belief space for the first and the second action, respectively.

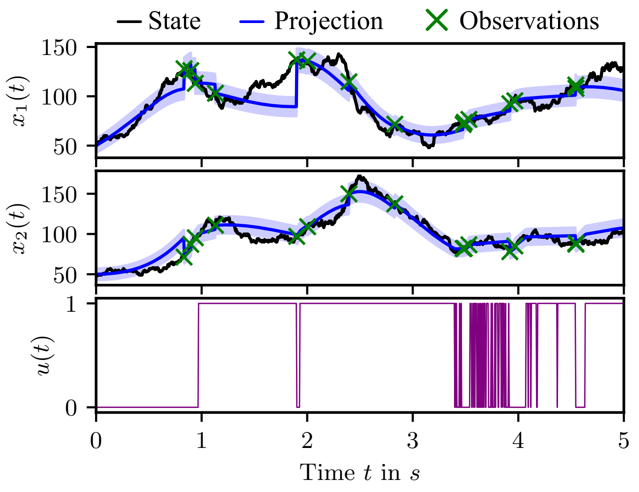

Figure 4: A sample trajectory of the LV problem using a policy computed by the QMPD method. The upper plots show the evolution of the exact states and the projection filter by indicating its mean and variance. The lower plot describes the actions over time.

Figure 4 depicts a sample trajectory of the controlled system. By comparison to samples with a constant control of either $u = 0$ or $u = 1$ (see Appendix 3), we see that the controlled trajectory is more stable in contrast to the oscillatory behaviour of the uncontrolled LV problem. This demonstrates that the controller effectively combines the dynamics of both actions, resulting in trajectories that are closer to the goal state.

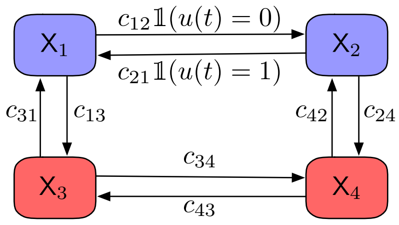

Figure 5: A schematic description of the considered CRN. Species X_3 and X_4 get observed exactly and are used to build an estimate of X_1 and X_2. The decision-maker can influence the flow between species X_1 and X_2.

5.3 Closed-Loop CRN

Finally, we use our method on a setup with sub-system continuous-time observations. We consider the $n = 4$ species CRN described in Fig. 5. The decision-maker controls the flow between species X_1 and X_2 with $u(t) = u$, where $u = 0$ corresponds to a flow from X_1 to X_2 and $u = 1$ corresponds to a flow from X_2 to X_1. The state of the system is described by $X(t) = [\hat{X}^\top(t), \bar{X}^\top(t)]^\top \in \mathcal{X}$, where we consider that the first two species are unobserved, i.e., $\hat{X}(t) = [X_1(t), X_2(t)]^\top$ and the last two species are observed, i.e., $Y(t) = \bar{X}(t) = [X_3(t), X_4(t)]^\top$. We design the reward function to favor a balanced system, where every species has the same number of molecules. For closed-loop networks the total species number N stays constant, therefore, we consider a filter approximation on $\mathcal{X} = \{0, 1, \dots, N\}^n$ using the multinomial family over the states $\hat{x}$, with $x = [\bar{x}^\top, \hat{x}^\top]^\top$ as

$$q_\theta(x) = \mathbb{1}_y(x)\,\mathrm{Mult}(\hat{x} \mid N - \bar{x}_1 - \bar{x}_2, \theta).$$

In Appendix 1.4 we derive closed-form solutions for the drift and jump updates of the multinomial parameters for general CRNs with sub-system measurements using the method of entropic matching.

Figure 6 shows a sample trajectory of the CRN and the corresponding evolution of the filtering distribution. We can see that the filtering distribution captures the behaviour of the latent states successfully. Also, the resulting policy leads to reasonable decisions. At the start of the trajectory, x_3 is the highest state while x_4 is the lowest. In response, the policy activates the flow from x_1 to x_2 to balance the state indirectly as a higher x_2 leads to a higher x_4 on average. In the second half of the trajectory the policy effectively switches between both actions to bring the system close to the goal state.

In Appendix 3 we provide additional experiments for systems with smaller state spaces, where exact filtering is tractable, to evaluate the effect the filtering approximation has on the control performance.

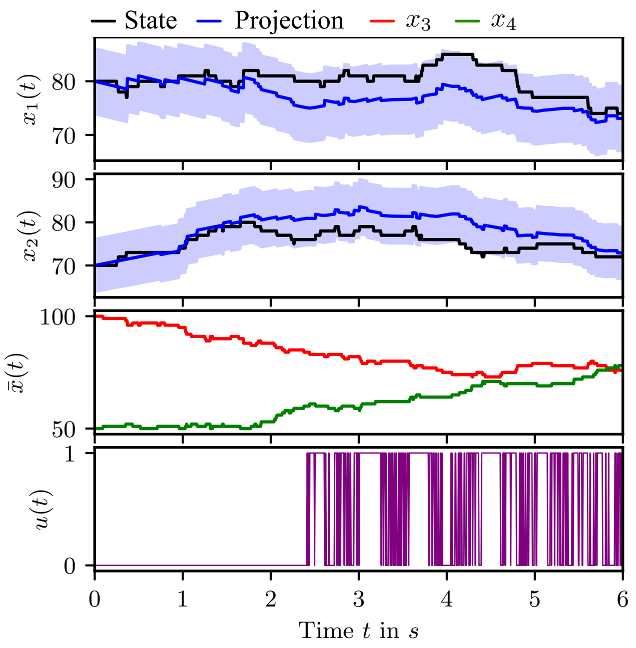

Figure 6: A sample trajectory of the CRN described in Fig. 5 using a policy computed by the QMDP method. The upper plots show the evolution of the exact states and the filtering distribution by indicating its mean and variance. The lower plots describe the observed species and the actions over time.

6 DISCUSSION

To the best of our knowledge, there exists no other work addressing POMDPs in continuous time with large discrete state spaces. While this limits direct comparisons with existing methods, we view this as a distinctive strength of our work, providing a solid foundation that future research can build upon.

As an alternative to the entropic matching method, we considered the use of sequential Monte Carlo methods (Doucet et al., 2001), such as the particle filter, which we compared in the Queueing problem. However, while sequential Monte Carlo methods may be a viable option for approximate filtering, integrating them into a control method in the considered problem setting is far from trivial. The primary challenge lies in the control policy's reliance on the belief generated by all samples, which necessitates frequent updates when any individual sample changes. While this update process is not an issue in discrete-time settings where all samples change simultaneously, it becomes impractically slow in continuous-time scenarios. In contrast, the control method based on the entropic matching approach is straightforward and circumvents this issue.

7 CONCLUSION

In this work, we presented the optimal control theory for continuous-time POMDPs with discrete state and action spaces. We discussed the arising difficulties for the filtering and the control problem when dealing with large state spaces. For both problems, we described approximations that scale well with an increasing state space. We then evaluated these methods on several partially observed systems.

In our experiments we approximated the filtering distributions with parametric distributions from the exponential family. While this choice allowed us to derive scalable closed-form solutions for the drift of the parameters, it is important to acknowledge that the expressiveness of these distributions has its limitations. Looking ahead, future research can explore more advanced parametric families to further enhance the capabilities of our approach. When it is not possible to obtain closed-form solutions, Monte Carlo methods offer an alternative approach to approximate the entropic matching equations. Moreover, we are interested in applying the presented methods on POMDPs, where the latent dynamics are described by stochastic differential equations. In that scenario, the belief is in general infinite dimensional and could be approximated by continuous parametric families of distributions. Also, this would enable the use for a broader range of applications including stochastic hybrid systems.

Acknowledgements

This work has been co-funded by the Distr@l-project BlueSwarm (Project 71585164) of the Hessian Ministry of Digital Strategy and Development and the German Research Foundation (DFG) within the Collaborative Research Center (CRC) 1053 MAKI.

References

B. Alt, M. Schultheis, and H. Koeppl. POMDPs in continuous time and discrete spaces. In *Advances in Neural Information Processing Systems*, volume 33, pages 13151–13162, 2020.

K. J. Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.

A. Bain and D. Crisan. *Fundamentals of stochastic filtering*, volume 3. Springer, 2009.

A. Bensoussan. *Stochastic control of partially observable systems.* Cambridge University Press, 1992.

D. Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 1. Athena Scientific, 2012a.

D. Bertsekas. *Dynamic programming and optimal control: Volume II*, volume 2. Athena Scientific, 2012b.

C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

G. Bolch, S. Greiner, H. De Meer, and K. S. Trivedi. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications.* John Wiley & Sons, 2006.

S. Bradtke and M. Duff. Reinforcement learning methods for continuous-time Markov decision problems. In *Advances in Neural Information Processing Systems*, volume 7, 1994.

D. Brigo, B. Hanzon, and F. Le Gland. Approximate nonlinear filtering by projection on exponential manifolds of densities. *Bernoulli*, pages 495–534, 1999.

L. Bronstein and H. Koeppl. Marginal process framework: A model reduction tool for Markov jump processes. *Physical Review E*, 97(6):062147, 2018a.

L. Bronstein and H. Koeppl. A variational approach to moment-closure approximations for the kinetics of biomolecular reaction networks. *The Journal of Chemical Physics*, 148(1), 2018b.

A. Doucet, N. de Freitas, and N. J. Gordon. *Sequential Monte Carlo methods in practice*, volume 1. Springer, 2001.

J. Du, J. Futoma, and F. Doshi-Velez. Model-based reinforcement learning for semi-Markov decision processes with neural ODEs. In *Advances in Neural Information Processing Systems*, volume 33, pages 19805–19816, 2020.

R. J. Elliott and W. P. Malcolm. General smoothing formulas for Markov-modulated Poisson observations. *IEEE Transactions on Automatic Control*, 50 (8):1123–1134, 2005.

S. N. Ethier and T. G. Kurtz. *Markov processes: characterization and convergence.* John Wiley & Sons, 2009.

C. Gardiner. *Stochastic methods*, volume 4. Springer, 2009.

L. Huang, L. Pauleve, C. Zechner, M. Unger, A. S. Hansen, and H. Koeppl. Reconstructing dynamic molecular states from single-cell time series. *Journal of The Royal Society Interface*, 13(122):20160533, 2016.

M. Igl, L. Zintgraf, T. A. Le, F. Wood, and S. Whiteson. Deep variational reinforcement learning for POMDPs. In *International Conference on Machine Learning*, pages 2117–2126, 2018.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

P. Karkus, D. Hsu, and W. S. Lee. QMDP-net: Deep learning for planning under partial observability. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

S. M. Kay. *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., 1993.

H. Kurniawati, D. Hsu, and W. S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Proceedings of Robotics: Science and Systems IV*, 2008.

H. J. Kushner. *Heavy traffic analysis of controlled queueing and communication networks*, volume 28. Springer, 2001.

M. Lauri, D. Hsu, and J. Pajarinen. Partially observable Markov decision processes in robotics: A survey. *IEEE Transactions on Robotics*, 2022.

M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning*, pages 362–370, 1995.

P. S. Maybeck. *Stochastic models, estimation, and control*. Academic Press, 1982.

T. Minka. Divergence measures and message passing. Technical Report MSR-TR-2005-173, Microsoft Research Ltd., 2005.

J. R. Norris. *Markov chains*. Cambridge University Press, 1998.

J. G. Proakis and M. Salehi. *Digital communications*. McGraw-Hill, fifth edition, 2008.

T. Ramalho, M. Selig, U. Gerland, and T. A. Enßlin. Simulation of stochastic network dynamics via entropic matching. *Phys. Rev. E*, 87:022719, 2013.

S. Särkkä. *Bayesian filtering and smoothing*. Cambridge University Press, 2013.

S. Särkkä and A. Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

D. Silver and J. Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems*, volume 23, 2010.

G. Singh, S. Peri, J. Kim, H. Kim, and S. Ahn. Structured world belief for reinforcement learning in POMDP. In *International Conference on Machine Learning*, pages 9744–9755, 2021.

A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

R. F. Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT Press, 2018.

C. Tallec, L. Blier, and Y. Ollivier. Making Deep Q-learning methods robust to time discretization. In *International Conference on Machine Learning*, pages 6096–6104, 2019.

S. Thrun. Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems*, volume 12, 1999.

D. J. Wilkinson. *Stochastic modelling for systems biology*. Chapman and Hall/CRC, 2018.

W. M. Wonham. On the separation theorem of stochastic control. *SIAM Journal on Control*, 6(2):312–326, 1968.

Q. Zhao, L. Tong, and A. Swami. Decentralized cognitive MAC for dynamic spectrum access. In *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005*, pages 224–232. IEEE, 2005.

E. Zhou, M. C. Fu, and S. I. Marcus. Solving continuous-state POMDPs via density projection. *IEEE Transactions on Automatic Control*, 55(5): 1101–1116, 2010.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes, see the end of Section 3.2]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]

   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Approximate Control for Continuous-Time POMDPs: Supplementary Materials

1 ENTROPIC MATCHING

1.1 General

In this subsection we give the derivation for the entropic matching equations Eqs. (8) and (10), following the steps in Bronstein and Koeppl (2018a,b).

As stated in Section 3.2, we start by assuming a parametric form for the filtering distribution at time point t as

$$\pi_t(x) \approx q_{\theta(t)}(x),$$

with parameters $\theta(t) \in \Theta \subseteq \mathbb{R}^p$. We then consider how the distribution $q_{\theta(t)}(x)$ evolves in a small time step h without new measurements. For the observation model D, the evolution between observations is described in Eq. (4), this yields

$$\tilde{\pi}_{t+h}(x) = q_{\theta(t)}(x) + h[\mathcal{L}_{u(t)}\, q_{\theta(t)}](x) + o(h).$$

The parameters of the distribution are computed for a small time step h by minimizing the reverse KL divergence

$$\theta(t+h) = \arg\min_{\theta'} \mathsf{KL}(\tilde{\pi}_{t+h} \parallel q_{\theta'}). \tag{15}$$

The idea of the entropic matching method is to approximate the change of the filtering distribution in an infinitesimal time step such that the approximated distribution stays in the space of the parametric distributions. Therefore we first set Appendix 1.1 in Eq. (15):

$$\begin{aligned}
&\mathsf{KL}(\tilde{\pi}_{t+h} \parallel q_{\theta'}) \\
&= \mathsf{KL}\big(q_{\theta(t)} + h[\mathcal{L}_{u(t)}\, q_{\theta(t)}] + o(h) \,\big\|\, q_{\theta'}\big) \\
&= \sum_{x\in\mathcal{X}} \big(q_{\theta(t)}(x) + h[\mathcal{L}_{u(t)}\, q_{\theta(t)}](x) + o(h)\big) \cdot \log \frac{q_{\theta(t)}(x) + h[\mathcal{L}_{u(t)}\, q_{\theta(t)}](x) + o(h)}{q_{\theta'}(x)} \\
&= \sum_{x\in\mathcal{X}} \left\{ q_{\theta(t)}(x) \log \frac{q_{\theta(t)}(x)}{q_{\theta'}(x)} + h \left[q_{\theta(t)}(x) \frac{[\mathcal{L}_{u(t)}\, q_{\theta(t)}](x)}{q_{\theta(t)}(x)} + [\mathcal{L}_{u(t)}\, q_{\theta(t)}](x) \log \frac{q_{\theta(t)}(x)}{q_{\theta'}(x)} \right] + o(h) \right\} \\
&= \mathsf{KL}\big(q_{\theta(t)} \parallel q_{\theta'}\big) + h\, \mathsf{E}_q \left[\frac{[\mathcal{L}_{u(t)}\, q_{\theta(t)}](X(t))}{q_{\theta(t)}(X(t))} + \frac{[\mathcal{L}_{u(t)}\, q_{\theta(t)}](X(t))}{q_{\theta(t)}(X(t))} \log \frac{q_{\theta(t)}(X(t))}{q_{\theta'}(X(t))} \right] + o(h),
\end{aligned}$$

where we used a first order Taylor series around $h = 0$ in the fourth line. The KL divergence between two members of a parametric distribution can be given by series expansion in $\theta - \theta'$ up to second order as

$$\mathsf{KL}(q_\theta \parallel q_{\theta'}) = \frac{1}{2}(\theta' - \theta)^\top F(\theta)(\theta' - \theta),$$

where $F(\theta) \coloneqq -\mathsf{E}_{q_\theta(x)}\left[\nabla_\theta \nabla_\theta^\top \log q_\theta(X)\right]$ is the Fisher information matrix. Hence, we can compute the minimum of Eq. (15) as

$$\begin{aligned}
0 &= \nabla_{\theta'} \mathsf{KL}(\tilde{\pi}_{t+h}(x) \parallel q_{\theta'}(x))|_{\theta'=\theta(t+h)} \\
&= F(\theta(t))(\theta(t+h) - \theta(t)) - h\, \mathsf{E}_q \left[\nabla_{\theta(t+h)} \log q_{\theta(t+h)}(X(t)) \frac{[\mathcal{L}_{u(t)}\, q_{\theta(t)}](X(t))}{q_{\theta(t)}(X(t))} \right] + o(h).
\end{aligned}$$

Dividing both sides by h and taking the limit $h \to 0$ we obtain

$$\begin{aligned}
0 &= F(\theta(t)) \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \mathsf{E}_q\left[\nabla_{\theta(t)} \log q_{\theta(t)}(X(t)) \frac{[\mathcal{L}_{u(t)} q_{\theta(t)}](X(t))}{q_{\theta(t)}(X(t))}\right] \\
&= F(\theta(t)) \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x q_{\theta(t)}(x) \nabla_{\theta(t)} \log q_{\theta(t)}(x) \frac{[\mathcal{L}_{u(t)} q_{\theta(t)}](x)}{q_{\theta(t)}(x)} \\
&= F(\theta(t)) \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x \nabla_{\theta(t)} \log q_{\theta(t)}(x) [\mathcal{L}_{u(t)} q_{\theta(t)}](x) \\
&= F(\theta(t)) \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x q_{\theta(t)}(x) [\mathcal{L}^\dagger_{u(t)} \nabla_{\theta(t)} \log q_{\theta(t)}](x) \\
&= F(\theta(t)) \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \mathsf{E}_q\left[\mathcal{L}^\dagger_{u(t)} \nabla_{\theta(t)} \log q_{\theta(t)}(X(t))\right].
\end{aligned}$$

This leads to the entropic matching equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) = F(\theta(t))^{-1} \mathsf{E}_q\left[\mathcal{L}^\dagger_{u(t)} \nabla_{\theta(t)} \log q_{\theta(t)}(X(t))\right].$$

For the sub-system measurement model C in Eq. (3), the evolution between between observation jumps is described in Eq. (10), this yields for the target filtering distribution

$$\tilde{\pi}_{t+h}(x) = q_{\theta(t)}(x) + h\, \mathbb{1}_{y(t)}(x) [\mathcal{L}_{u(t)} q_{\theta(t)}](x) - h q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x') + o(h).$$

Setting this in Eq. (15) leads to:

$$\begin{aligned}
&\mathsf{KL}(\tilde{\pi}_{t+h} \parallel q_{\theta'}) \\
&= \sum_{x \in \mathcal{X}} \left(q_{\theta(t)}(x) + h\, \mathbb{1}_{y(t)}(x) [\mathcal{L}_{u(t)} q_{\theta(t)}](x) - h q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x') + o(h) \right) \\
&\qquad \cdot \left[\log \frac{q_{\theta(t)}(x) + h\, \mathbb{1}_{y(t)}(x) [\mathcal{L}_{u(t)} q_{\theta(t)}](x) - h q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x') + o(h)}{q_{\theta'}(x)} \right] \\
&= \sum_{x \in \mathcal{X}} \Bigg\{ q_{\theta(t)}(x) \log \frac{q_{\theta(t)}(x)}{q_{\theta'}(x)} \\
&\qquad + h \left[q_{\theta(t)}(x) \frac{\mathbb{1}_{y(t)}(x) [\mathcal{L}_{u(t)} q_{\theta(t)}](x) - q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x')}{q_{\theta(t)}(x)} \right. \\
&\qquad + \left. \left(\mathbb{1}_{y(t)}(x) [\mathcal{L}_{u(t)} q_{\theta(t)}](x) - q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x') \right) \log \frac{q_{\theta(t)}(x)}{q_{\theta'}(x)} \right] + o(h) \Bigg\} \\
&= \mathsf{KL}\big(q_{\theta(t)} \parallel q_{\theta'}\big) \\
&\qquad + h\, \mathsf{E}_q \left[\frac{\mathbb{1}_{y(t)}(X(t)) [\mathcal{L}_{u(t)} q_{\theta(t)}](X(t)) - q_{\theta(t)}(X(t)) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x')}{q_{\theta(t)}(X(t))} \right. \\
&\qquad + \left. \frac{\mathbb{1}_{y(t)}(X(t)) [\mathcal{L}_{u(t)} q_{\theta(t)}](X(t)) - q_{\theta(t)}(X(t)) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x')}{q_{\theta(t)}(X)} \log \frac{q_{\theta(t)}(X(t))}{q_{\theta'}(X(t))} \right] + o(h),
\end{aligned}$$

where we used a first order Taylor series around $h = 0$ in the third line.

By the same argument as before, we can compute the minimum of Eq. (15) as

$$\begin{aligned}
0 &= \nabla_{\theta'} \mathsf{KL}(\tilde{\pi}_{t+h}(x) \parallel q_{\theta'}(x))|_{\theta' = \theta(t+h)} \\
&= F(\theta(t))(\theta(t+h) - \theta(t)) \\
&\quad - h\, \mathsf{E}_q \left[\nabla_{\theta(t+h)} \log q_{\theta(t+h)}(X(t)) \frac{\mathbb{1}_{y(t)}(X(t)) [\mathcal{L}_{u(t)} q_{\theta(t)}](X(t)) - q_{\theta(t)}(X(t)) \sum_{x'} \mathbb{1}_{y(t)}(x') [\mathcal{L}_{u(t)} q_{\theta(t)}](x')}{q_{\theta(t)}(X(t))} \right] + o(h)
\end{aligned}$$

Dividing both sides by h and taking the limit $h \to 0$ we obtain

$$\begin{aligned}
0 &= F(\theta(t))\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \mathsf{E}_q\left[\nabla_{\theta(t)} \log q_{\theta(t)}(X(t)) \frac{\mathbb{1}_{y(t)}(X(t))[\mathcal{L}_{u(t)} q_{\theta(t)}](X(t)) - q_{\theta(t)}(X(t)) \sum_{x'} \mathbb{1}_{y(t)}(x')[\mathcal{L}_{u(t)} q_{\theta(t)}](x')}{q_{\theta(t)}(X(t))}\right] \\
&= F(\theta(t))\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x q_{\theta(t)}(x) \nabla_{\theta(t)} \log q_{\theta(t)}(x) \frac{\mathbb{1}_{y(t)}(x)[\mathcal{L}_{u(t)} q_{\theta(t)}](x) - q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x')[\mathcal{L}_{u(t)} q_{\theta(t)}](x')}{q_{\theta(t)}(x)} \\
&= F(\theta(t))\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x \nabla_{\theta(t)} \log q_{\theta(t)}(x) \left(\mathbb{1}_{y(t)}(x)[\mathcal{L}_{u(t)} q_{\theta(t)}](x) - q_{\theta(t)}(x) \sum_{x'} \mathbb{1}_{y(t)}(x')[\mathcal{L}_{u(t)} q_{\theta(t)}](x')\right) \\
&= F(\theta(t))\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x \nabla_{\theta(t)} \log q_{\theta(t)}(x) \left(\mathbb{1}_{y(t)}(x)[\mathcal{L}_{u(t)} q_{\theta(t)}](x)\right) \\
&= F(\theta(t))\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \sum_x q_{\theta(t)}(x) \mathcal{L}^{\dagger}_{u(t)} \left\{\mathbb{1}_{y(t)} \cdot \nabla_{\theta(t)} \log q_{\theta(t)}\right\}(x) \\
&= F(\theta(t))\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) - \mathsf{E}_q\left[\mathcal{L}^{\dagger}_{u(t)} \left\{\mathbb{1}_{y(t)} \cdot \nabla_{\theta(t)} \log q_{\theta(t)}\right\}(X(t))\right],
\end{aligned}$$

where in the fourth line we used $\mathsf{E}_q\left[\nabla_{\theta(t)} \log q_{\theta(t)}(X(t))\right] = 0$. This leads to the entropic matching equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) = F(\theta(t))^{-1}\, \mathsf{E}_q\left[\mathcal{L}^{\dagger}_{u(t)} \left\{\mathbb{1}_{y(t)} \cdot \nabla_{\theta(t)} \log q_{\theta(t)}\right\}(X(t))\right].$$

1.2 Entropic Matching for Finite Buffer Queues

We consider a queueing network with n queues. We assumed that the $i$the queue is an $M/M/c/n$ queue, with Markovian arrivals and services, a number of c servers and a finite buffer of size n, for more, see (Bolch et al., 2006). The state of the queueing network is described by the number of packets in the queues, i.e., $x = [x_1, \dots, x_n]^\top$, with $x_i \in \mathcal{X}_i \coloneqq \{0, 1, \dots, N_i\}$ and $\mathcal{X} \coloneqq \bigtimes_{i=1}^n \mathcal{X}_i$. We denote the stochastic routing matrix by $\mathbf{P} \in \Delta^{n+1 \times n+1}$. The entry P_{ij} denotes the probability of routing a packet from queue i to queue j, with $(i,j) \in \{1, \dots n\} \times \{1, \dots, n\}$. Additionally, we denote by the entries $P_{i,n+1}$ and $P_{n+1,i}$ the probabilities that a packet leaves and enters the queueing network from and into the queue i, respectively. The corresponding arrival, and respectively service, rates are denoted by $\{\tilde{\lambda}_{ij}\}$. The effective rate for a number of c_i servers is then given by

$$\lambda_{ij}(x_i) = \tilde{\lambda}_{ij} P_{ij} \min(x_i, c_i),$$

where we set $x_{n+1} = c_{n+1} = 1$ and $\mathcal{X}_{n+1} = \mathbb{N}_0$ for convenience. We define the corresponding jump vectors as

$$\nu_{ij} \coloneqq e_j - e_i,$$

where e_i is the ith unit vector, and we set by definition e_{n+1} to the n-dimensional zero vector. The system is then described by a continuous-time Markov chain $\{X(t)\}_{t \in \mathbb{R}_{\geq 0}}$, with reaction rate function $\Lambda(x, x') \coloneqq \lim_{h \to 0} h^{-1}\, \mathrm{P}(X(t+h) = x' \mid X(t) = x)$ given as

$$\Lambda(x, x') = \mathbb{1}(x' \in \mathcal{X})\, \mathbb{1}(x \in \mathcal{X}) \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i)\, \mathbb{1}(x' = x + \nu_{ij}),$$

for all $(x, x') \in \mathcal{X} \times \mathcal{X}$. The forward master equation is therefore given by

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} p_t(x) &= [\mathcal{L} p_t](x) \\
&= \sum_{x'} \Lambda(x', x) p_t(x') \\
&= \sum_{x' \neq x} \Lambda(x', x) p_t(x') - \sum_{x' \neq x} \Lambda(x, x') p_t(x) \\
&= \sum_{x' \neq x} \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x'_i)\, \mathbb{1}(x = x' + \nu_{ij}) p_t(x') - \sum_{x' \neq x} \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i)\, \mathbb{1}(x' = x + \nu_{ij}) p_t(x) \\
&= \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i + 1)\, \mathbb{1}(x - \nu_{ij} \in \mathcal{X}) p_t(x - \nu_{ij}) - \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i)\, \mathbb{1}(x + \nu_{ij} \in \mathcal{X}) p_t(x) \\
\frac{\mathrm{d}}{\mathrm{d}t} p_t(x) &= \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i + 1)\, \mathbb{1}(x_i + 1 \in \mathcal{X}_i)\, \mathbb{1}(x_j - 1 \in \mathcal{X}_j) p_t(x - \nu_{ij}) \\
&\quad - \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i)\, \mathbb{1}(x_i - 1 \in \mathcal{X}_i)\, \mathbb{1}(x_j + 1 \in \mathcal{X}_j) p_t(x).
\end{aligned}$$

Let, $\mathcal{L}^\dagger$ denote the adjoint operator of $\mathcal{L}$, w.r.t, the inner product $\langle \phi, \psi \rangle \coloneqq \sum_x \phi(x)\psi(x)$, i.e,

$$\begin{aligned}
&\langle \mathcal{L}\phi, \psi \rangle = \langle \phi, [\mathcal{L}^\dagger \psi] \rangle \\
\iff &\sum_x [\mathcal{L}\phi](x) \psi(x) = \langle \phi, [\mathcal{L}^\dagger \psi] \rangle \\
\iff &\sum_x \sum_{x'} \Lambda(x', x) \phi(x') \psi(x) = \langle \phi, [\mathcal{L}^\dagger \psi] \rangle \\
\iff &\sum_{x'} \phi(x') \sum_x \Lambda(x', x) \psi(x) = \langle \phi, [\mathcal{L}^\dagger \psi] \rangle.
\end{aligned}$$

Hence, we have

$$[\mathcal{L}^\dagger \phi](x) = \sum_{x'} \Lambda(x, x') \phi(x').$$

Therefore, we can find the adjoint operator $\mathcal{L}^\dagger$ working on a test function ϕ as

$$\begin{aligned}
[\mathcal{L}^\dagger \phi](x) &= \sum_{x'} \Lambda(x, x') \phi(x') \\
&= \sum_{x' \neq x} \Lambda(x, x') \phi(x') - \sum_{x' \neq x} \Lambda(x, x') \phi(x) \\
&= \sum_{x' \neq x} \Lambda(x, x') (\phi(x') - \phi(x)) \\
&= \sum_{x' \neq x} \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i)\, \mathbb{1}(x' = x + \nu_{ij}) (\phi(x') - \phi(x)) \\
[\mathcal{L}^\dagger \phi](x) &= \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \lambda_{ij}(x_i)\, \mathbb{1}(x_i - 1 \in \mathcal{X}_i)\, \mathbb{1}(x_j + 1 \in \mathcal{X}_j) (\phi(x + \nu_{ij}) - \phi(x)).
\end{aligned}$$

Let, $\phi(x) = \nabla \log q_\theta(x)$ and $q_\theta(x) = \prod_{i=1}^n \mathrm{Bin}(x_i \mid N_i, \theta_i)$, we compute

$$\partial_{\theta_i} \log q_\theta(x) = \frac{x_i - N_i \theta_i}{\theta_i (1 - \theta_i)}.$$

The inverse Fisher information matrix is given by

$$F(\theta)^{-1} = \operatorname{diag}\left(\left[\frac{\theta_1(1-\theta_1)}{N_1}, \dots, \frac{\theta_n(1-\theta_n)}{N_n}\right]^\top\right).$$

Note that,

$$\nabla_\theta \log q_\theta(x+\nu_{ij}) - \nabla_\theta \log q_\theta(x) = \frac{e_j}{\theta_j(1-\theta_j)} - \frac{e_i}{\theta_i(1-\theta_i)}.$$

Therefore we have for the entropic matching equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta = F(\theta)^{-1}\,\mathsf{E}_q[\mathcal{L}^\dagger \nabla_\theta \log q_\theta(X)],$$

the ODE

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) &= \mathsf{E}_q\left[\sum_{i=1}^{n+1}\sum_{j=1}^{n+1} \lambda_{ij}(X_i)\,\mathbb{1}(X_i - 1 \in \mathcal{X}_i)\,\mathbb{1}(X_j + 1 \in \mathcal{X}_j)(\frac{e_j}{N_j} - \frac{e_i}{N_i})\right] \\ &= \sum_{i=1}^{n+1}\sum_{j=1}^{n+1}(\frac{e_j}{N_j} - \frac{e_i}{N_i})\,\mathsf{E}_q\left[\lambda_{ij}(X_i)\,\mathbb{1}(X_i - 1 \in \mathcal{X}_i)\right]\mathsf{E}[\mathbb{1}(X_j + 1 \in \mathcal{X}_j)]\,.\end{aligned}$$

This can be written component-wise as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\theta_i(t) =& \frac{1}{N_i}\sum_{j=1}^{n+1}\left(\mathsf{E}_q[\lambda_{ji}(X_j)\,\mathbb{1}(X_j - 1 \in \mathcal{X}_j)]\,\mathsf{E}_q[\mathbb{1}(X_i + 1 \in \mathcal{X}_i)]\right. \\ &\left. - \mathsf{E}_q[\lambda_{ij}(X_i)\,\mathbb{1}(X_i - 1 \in \mathcal{X}_i)]\,\mathsf{E}_q[\mathbb{1}(X_j + 1 \in \mathcal{X}_j)]\right).\end{aligned}$$

We compute for $j \in \{1, \dots, n\}$

$$\begin{aligned}\mathsf{E}_q[\mathbb{1}(X_j + 1 \in \mathcal{X}_j)] &= \sum_{x_j=0}^{N_j} \operatorname{Bin}(x_j \mid N_j, \theta_j)\,\mathbb{1}(x_j + 1 \in \mathcal{X}_j) \\ &= 1 - \sum_{x_j=0}^{N_j} \operatorname{Bin}(x_j \mid N_j, \theta_j)\,\mathbb{1}(x_j + 1 \notin \mathcal{X}_j) \\ &= 1 - \operatorname{Bin}(N_j \mid N_j, \theta_j)\end{aligned}$$

and for $j = n+1$

$$\mathsf{E}_q[\mathbb{1}(X_j + 1 \in \mathcal{X}_j)] = 1.$$

For $i \in \{1, \dots, n\}$

$$\begin{aligned}\mathsf{E}_q[\lambda_{ij}(X_i)\,\mathbb{1}(X_i - 1 \in \mathcal{X}_i)] &= \sum_{x_i=0}^{N_i} \operatorname{Bin}(x_i \mid N_i, \theta_i)\tilde{\lambda}_{ij} P_{ij} \min(x_i, c_i)\,\mathbb{1}(x_i - 1 \in \mathcal{X}_i) \\ &= \tilde{\lambda}_{ij} P_{ij}(\sum_{x_i=0}^{c_i-1} \operatorname{Bin}(x_i \mid N_i, \theta_i) x_i + \sum_{x_i=c_i}^{N_i} \operatorname{Bin}(x_i \mid N_i, \theta_i) c_i) \\ &= \tilde{\lambda}_{ij} P_{ij}(\sum_{x_i=0}^{c_i-1} \operatorname{Bin}(x_i \mid N_i, \theta_i) x_i + c_i(1 - \sum_{x_i=0}^{c_i-1} \operatorname{Bin}(x_i \mid N_i, \theta_i)) \\ &= \tilde{\lambda}_{ij} P_{ij}(c_i + \sum_{x_i=0}^{c_i-1} \operatorname{Bin}(x_i \mid N_i, \theta_i)(x_i - c_i))\end{aligned}$$

and for $i = n+1$ we have

$$\mathsf{E}_q[\lambda_{ij}(X_i)\,\mathbb{1}(X_i - 1 \in \mathcal{X}_i)] = \tilde{\lambda}_{ij} P_{ij}.$$

Therefore, we have the component-wise ODE

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\theta_i(t) &= \frac{1}{N_i}\sum_{j=1}^{n+1} \mathsf{E}_q[\lambda_{ji}(X_j)\,\mathbb{1}(X_j-1\in\mathcal{X}_j)]\,\mathsf{E}_q[\mathbb{1}(X_i+1\in\mathcal{X}_i)] \\
&\quad - \mathsf{E}_q[\lambda_{ij}(X_i)\,\mathbb{1}(X_i-1\in\mathcal{X}_i)]\,\mathsf{E}_q[\mathbb{1}(X_j+1\in\mathcal{X}_j)] \\
&= \frac{1}{N_i}\sum_{j=1}^{n}(\tilde\lambda_{ji}P_{ji}(c_j+\sum_{x_j=0}^{c_j-1}\mathrm{Bin}(x_j\mid N_j,\theta_j)(x_j-c_j))(1-\mathrm{Bin}(N_i\mid N_i,\theta_i)) \\
&\quad - \tilde\lambda_{ij}P_{ij}(c_i+\sum_{x_i=0}^{c_i-1}\mathrm{Bin}(x_i\mid N_i,\theta_i)(x_i-c_i))(1-\mathrm{Bin}(N_j\mid N_j,\theta_j)) \\
&\quad + \frac{1}{N_i}\tilde\lambda_{n+1,i}P_{n+1,i}(1-\mathrm{Bin}(N_i\mid N_i,\theta_i)) - \frac{1}{N_i}\tilde\lambda_{i,n+1}P_{i,n+1}(c_i+\sum_{x_i=0}^{c_i-1}\mathrm{Bin}(x_i\mid N_i,\theta_i)(x_i-c_i)) \\
&= \frac{1-\mathrm{Bin}(N_i\mid N_i,\theta_i)}{N_i}(\tilde\lambda_{n+1,i}P_{n+1,i}+\sum_{j=1}^{n}\tilde\lambda_{ji}P_{ji}(c_j+\sum_{x_j=0}^{c_j-1}\mathrm{Bin}(x_j\mid N_j,\theta_j)(x_j-c_j))) \\
&\quad - \frac{c_i+\sum_{x_i=0}^{c_i-1}\mathrm{Bin}(x_i\mid N_i,\theta_i)(x_i-c_i)}{N_i}(\tilde\lambda_{i,n+1}P_{i,n+1}+\sum_{j=1}^{n}\tilde\lambda_{ij}P_{ij}(1-\mathrm{Bin}(N_j\mid N_j,\theta_j)))
\end{aligned}$$

In the example we assume that the number of servers is set to $c_1 = c_2 = c_3 = 1$. Note that, we have $\mathrm{Bin}(0\mid N_i,\theta_i) = (1-\theta_i)^{N_i}$ and $\mathrm{Bin}(N_i\mid N_i,\theta_i) = \theta_i^{N_i}$, hence,

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_i = \frac{1-\theta_i^{N_i}}{N_i}(\tilde\lambda_{n+1,i}P_{n+1,i}+\sum_{j=1}^{n}\tilde\lambda_{ji}P_{ji}(1-(1-\theta_j)^{N_j}) - \frac{1-(1-\theta_i)^{N_i}}{N_i}(\tilde\lambda_{i,n+1}P_{i,n+1}+\sum_{j=1}^{n}\tilde\lambda_{ij}P_{ij}(1-\theta_j^{N_j})).$$

In the experiments we use a queueing network with $n = 3$, queues with equal buffer size $N = N_1 = N_2 = N_3$ and We set the routing matrix to

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & \mathbb{1}(u(t)=0) & 0 \\ 0 & 0 & \mathbb{1}(u(t)=1) & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}.$$

At observation time points we update the parameters accordingly as

$$\theta(t_i) = \arg\min_{\theta'} \mathsf{KL}\Big(\pi_{t_i\mid\theta(t_i^-)}\,\Big\|\,q_{\theta'}\Big),$$

where the KL divergence is computed w.r.t.

$$\pi_{t_i\mid\theta(t_i^-)}(x) \propto p(y_i\mid x, u(t_i^-))\,q_{\theta(t_i^-)}(x)).$$

For members of the exponential family parameterization as

$$q_\theta(x) = q_0(x)\exp(\theta^\top T(x) - A(\theta)),$$

with base measure $q_0(x)$, natural parameters θ, sufficient statistics $T(x)$, and log-normalizer $A(\theta)$, the optimization at the discrete observation time points, reduces to the problem of moment matching (Bishop, 2006), i.e.,

$$\mathsf{E}_q[T(X)] = \mathsf{E}_\pi[T(X)].$$

While this can be computed exact for examples with small state spaces, it can be approximated with Monte Carlo samples for large state spaces. In our experiments we use a different approximation for the update in order to obtain closed-form solutions. For queues with a large buffer size N_i, we approximate the binomial distribution at observation times by a Gaussian distribution via matching the moments, i.e., $\mathcal{N}(x_i\mid N_i\theta_i, N_i\theta_i(1-\theta_i))$. Since we consider Gaussian measurements, we can compute the posterior distribution and again match the moments to approximate the posterior by a binomial distribution.

1.3 Entropic Matching for Chemical Reaction Networks using a Product Poisson Distribution

Chemical reaction networks (Wilkinson, 2018) are a subclass of CTMCs defined as a system of k reactions involving n species as

$$\begin{gathered} S_{11}\mathsf{X}_1 + \cdots + S_{n1}\mathsf{X}_n \xrightarrow{c_1} P_{11}\mathsf{X}_1 + \cdots + P_{n1}\mathsf{X}_n \\ \vdots \\ S_{1k}\mathsf{X}_1 + \cdots + S_{nk}\mathsf{X}_n \xrightarrow{c_k} P_{1k}\mathsf{X}_1 + \cdots + P_{nk}\mathsf{X}_n, \end{gathered}$$

where $c_j \in \mathbb{R}_{\geq 0}$ is called the reaction rate of the j-th reaction and $S_{ij} \in \mathbb{N}_0$ and $P_{ij} \in \mathbb{N}_0$ are the stoichiometric substrate and product coefficients for species X_i in the j-th reaction, respectively. The state of the network is described by the size of each species, i.e., $x = [x_1, \ldots, x_n]^\top$, with $x_i \in \mathcal{X}_i \coloneqq \mathbb{N}_0$ and $\mathcal{X} \coloneqq \bigtimes_{i=1}^n \mathcal{X}_i$. The change vector $v_j \in \mathbb{Z}^n$ corresponding to the j-th reaction is defined by

$$v_j = \begin{pmatrix} P_{1j} - S_{1j} \\ \vdots \\ P_{nj} - S_{nj} \end{pmatrix},$$

and the propensity corresponding to the j-th reaction is given by mass-action kinetics as

$$\lambda_j(x, u) = c_j(u) \prod_{i=1}^n \binom{x_i}{S_{ij}}.$$

Note that we add the possibility to control the CRN by making the reaction coefficients action dependent. From this we can define the rate function of the CTMC as

$$\Lambda(x, x', u) = \sum_{j=1}^k \mathbb{1}(x' = x + v_j)\lambda_j(x, u).$$

We now want to approximate the filtering distribution of a CRN with discrete time observations by a product Poisson distribution as

$$q_\theta(x) = \prod_{i=1}^n \mathrm{Pois}(x_i \mid \theta_i).$$

Using the entropic matching method the evolution of the parameters can be described by

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) = F(\theta(t))^{-1}\, \mathsf{E}_q\left[\mathcal{L}^\dagger_{u(t)} \nabla_{\theta(t)} \log q_{\theta(t)}(X(t))\right].$$

For a set of reactions the adjoint operator of the evolution operator which acts on functions ψ is given by

$$[\mathcal{L}^\dagger_u \psi](x) = \sum_{j=1}^k \lambda_j(x, u)\{\psi(x + v_j) - \psi(x)\}.$$

As $\nabla_{\theta(t)} \log q_{\theta(t)}(x)$ is linear in x, the term inside the expectation can be reduced to

$$[\mathcal{L}^\dagger_{u(t)} \nabla_{\theta(t)} \log q_{\theta(t)}](x) = \sum_{j=1}^k \frac{v_j}{\theta} \lambda_j(x, u(t)).$$

Combining this with the Fisher matrix of the product Poisson distribution the drift of the lth parameter simplifies to

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_l(t) = \theta_l(t)\, \mathsf{E}_q\left[\frac{1}{\theta_l(t)} \sum_{j=1}^k \lambda_j(X(t), u(t)) v_{lj}\right]$$

$$
\begin{aligned}
&= \sum_{j=1}^{k} \mathsf{E}_q \left[\lambda_j(X(t), u(t))\right] v_{lj} \\
&= \sum_{j=1}^{k} \mathsf{E}_q \left[c_j(u(t)) \prod_{i=1}^{n} \binom{X_i(t)}{S_{ij}} \right] v_{lj} \\
&= \sum_{j=1}^{k} c_j(u(t)) \prod_{i=1}^{n} \mathsf{E}_q \left[\frac{X_i(t)!}{S_{ij}!(X_i(t) - S_{ij})!} \right] v_{lj} \\
&= \sum_{j=1}^{k} c_j(u(t)) \prod_{i=1}^{n} \frac{\theta_i^{S_{ij}}(t)}{S_{ij}!} v_{lj}.
\end{aligned}
$$

For the LV problem considered in the experiments we get the following drift:

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\theta_1(t) &= c_1\theta_1(t) - c_2\theta_1(t)\theta_2(t) + c_1 \\
\frac{\mathrm{d}}{\mathrm{d}t}\theta_2(t) &= c_2\theta_1(t)\theta_2(t) - c_3(u(t))\theta_2(t).
\end{aligned}
$$

Combining the drift in between measurements with the moment matching method at the time points new measurements are given, we can describe the evolution of the parameters in time.

Similar to the entropic matching for finite buffer queues using a product binomial distribution, we can compute the update at observation time points using moment matching, which can be approximated with Monte Carlo samples. In our experiments we use a different approximation for the update in order to obtain closed-form solutions. We approximate the Poisson distribution at observation times by a Gaussian distribution via matching the moments, i.e., $\mathcal{N}(x_i \mid \theta_i, \theta_i)$. Since we consider Gaussian measurements, we can compute the posterior distribution and again match the moments to approximate the posterior by a Poisson distribution.

1.4 Entropic Matching for Chemical Reaction Networks with Sub-System Measurements using a Multinomial Distribution

Consider CRNs with $n = \hat{n} + \bar{n}$ species. We denote the state of all species in the system as $x = [\hat{x}^\top, \bar{x}^\top]^\top$, where the first $\hat{n}$ states, also denoted by $\hat{x} = [x_1, \ldots, x_{\hat{n}}]^\top$, are being estimated using exact continuous observations of the latter $\bar{n}$ states $\bar{x} = [x_{\hat{n}+1}, x_{\hat{n}+2}, \ldots, x_{\hat{n}+\bar{n}}]^\top$. The exact filter for these systems is considered in (Bronstein and Koeppl, 2018a).

We want to approximate the exact filter using a multinomial distribution as

$$
q_\theta(x) = \mathbb{1}_{y(t)}(x) q_\theta(\hat{x}) = \mathbb{1}_{y(t)}(x) \operatorname{Mult}(\hat{x} \mid N - \sum_{i=1}^{\bar{n}} \bar{x}_i, \theta),
$$

with event probabilities θ and number of trials $\hat{N} = N - \sum_{i=1}^{\bar{n}} \bar{x}_i$.

Using the method of entropic matching we compute the evolution of the parameters as

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) &= F(\theta(t))^{-1} \mathsf{E}_q \left[\mathcal{L}_{u(t)}^{\dagger} \{ \mathbb{1}_{y(t)} \cdot \nabla_{\theta(t)} \log q_{\theta(t)} \}(X(t)) \right] \\
&= F(\theta(t))^{-1} \mathsf{E}_q \left[\sum_{j=1}^{R} h_j(X(t)) \{ \mathbb{1}_{y(t)}(X(t) + v_j) \nabla_{\theta(t)} \log q_{\theta(t)}(X(t) + v_j) \right. \\
&\qquad \left. - \mathbb{1}_{y(t)}(X(t)) \nabla_{\theta(t)} \log q_{\theta(t)}(X(t)) \} \right] \\
&= F(\theta(t))^{-1} \mathsf{E}_q \left[\sum_{j=1}^{R} h_j(X(t)) \{ \mathbb{1}(\bar{v}_j = 0)(\nabla_{\theta(t)} \log q_{\theta(t)}(X(t) + v_j) - \nabla_{\theta(t)} \log q_{\theta(t)}(X(t))) \right. \\
&\qquad \left. - \mathbb{1}(\bar{v}_j \neq 0) \nabla_{\theta(t)} \log q_{\theta(t)}(X(t)) \} \right]
\end{aligned}
$$

$$
\begin{aligned}
&= F(\theta(t))^{-1} \sum_{j=1}^{R} c_j \prod_{i=1}^{\bar{n}} \binom{y_i}{\bar{S}_{ij}} \\
&\quad \left(\mathbb{1}(\bar{v}_j = 0)\, \mathsf{E}_q \left[\prod_{i=1}^{\hat{n}} \binom{\hat{X}_i(t)}{\hat{S}_{ij}} (\nabla_{\theta(t)} \log q_{\theta(t)}(X(t)+v_j) - \nabla_{\theta(t)} \log q_{\theta(t)}(X(t))) \right] \right. \\
&\quad \left. - \mathbb{1}(\bar{v}_j \neq 0)\, \mathsf{E}_q \left[\prod_{i=1}^{\hat{n}} \binom{\hat{X}_i(t)}{\hat{S}_{ij}} \nabla_{\theta(t)} \log q_{\theta(t)}(X(t)) \right] \right),
\end{aligned}
$$

where the gradient of the log probability is given by

$$
\nabla_{\theta(t)} \log q_{\theta(t)}(x) = \mathbb{1}_{y(t)}(x) \begin{pmatrix} \frac{\hat{x}_1}{\theta_1(t)} - \frac{\hat{x}_{\hat{n}}}{\theta_{\hat{n}}(t)} \\ \vdots \\ \frac{\hat{x}_{\hat{n}-1}}{\theta_{\hat{n}-1}(t)} - \frac{\hat{x}_{\hat{n}}}{\theta_{\hat{n}}(t)} \end{pmatrix}.
$$

This leads to

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) &= F(\theta(t))^{-1} \sum_{j=1}^{R} c_j \prod_{i=1}^{\bar{n}} \binom{y_i}{\bar{S}_{ij}} \\
&\quad \left(\mathbb{1}(\bar{v}_j = 0)\, \mathsf{E}_q \left[\prod_{i=1}^{\hat{n}} \binom{\hat{X}_i(t)}{\hat{S}_{ij}} \right] \begin{pmatrix} \frac{\hat{v}_{1j}}{\theta_1(t)} - \frac{\hat{v}_{\hat{n}j}}{\theta_{\hat{n}}(t)} \\ \vdots \\ \frac{\hat{v}_{\hat{n}-1,j}}{\theta_{\hat{n}-1}(t)} - \frac{\hat{v}_{\hat{n}j}}{\theta_{\hat{n}}(t)} \end{pmatrix} - \mathbb{1}(\bar{v}_j \neq 0)\, \mathsf{E}_q \left[\prod_{i=1}^{\hat{n}} \binom{\hat{X}_i(t)}{\hat{S}_{ij}} \begin{pmatrix} \frac{\hat{X}_1(t)}{\theta_1(t)} - \frac{\hat{X}_{\hat{n}}(t)}{\theta_{\hat{n}}(t)} \\ \vdots \\ \frac{\hat{X}_{\hat{n}-1}(t)}{\theta_{\hat{n}-1}(t)} - \frac{\hat{X}_{\hat{n}}(t)}{\theta_{\hat{n}}(t)} \end{pmatrix} \right] \right).
\end{aligned}
$$

The expectations for the multinomial distribution are given by

$$
\mathsf{E}_q \left[\prod_{i=1}^{\hat{n}} \binom{\hat{X}_i}{\hat{S}_{ij}} \right] = \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!},
$$

$$
\mathsf{E}_q \left[\prod_{i=1}^{\hat{n}} \binom{\hat{X}_i}{\hat{S}_{ij}} \hat{X}_l \right] = \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!} \left(\hat{S}_{lj} + \theta_l (N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij}) \right).
$$

Inserting these expectations, we get the following for the drift:

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\theta(t) =& F(\theta(t))^{-1} \sum_{j=1}^{R} c_j \prod_{i=1}^{\bar{n}} \binom{y_i}{\bar{S}_{ij}} \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}(t)}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!} \\
& \left(\mathbb{1}(\bar{v}_j = 0) \begin{pmatrix} \frac{\hat{v}_{1j}}{\theta_1(t)} - \frac{\hat{v}_{\hat{n}j}}{\theta_{\hat{n}}(t)} \\ \vdots \\ \frac{\hat{v}_{\hat{n}-1,j}}{\theta_{\hat{n}-1}(t)} - \frac{\hat{v}_{\hat{n}j}}{\theta_{\hat{n}}(t)} \end{pmatrix} - \mathbb{1}(\bar{v}_j \neq 0) \begin{pmatrix} \frac{\hat{S}_{1j}}{\theta_1(t)} - \frac{\hat{S}_{\hat{n}j}}{\theta_{\hat{n}}(t)} \\ \vdots \\ \frac{\hat{S}_{\hat{n}-1,j}}{\theta_{\hat{n}-1}(t)} - \frac{\hat{S}_{\hat{n}j}}{\theta_{\hat{n}}(t)} \end{pmatrix} \right).
\end{aligned}
$$

The fisher matrix and its inverse are given by

$$
F(\theta)_{ij} = \frac{\hat{N}}{\theta_{\hat{n}}} + \delta_{ij} \frac{\hat{N}}{\theta_i}
$$

$$
F(\theta)^{-1}_{ij} = \frac{1}{\hat{N}^2} \mathrm{Cov}_{ij},
$$

where Cov_{ij} are the elements of the covariance matrix of the multinomial distribution for $1 \leq i, j \leq \hat{n}-1$. This leads to the drift:

$$\begin{aligned}
\frac{\mathrm{d}\theta_l(t)}{\mathrm{d}t} &= \sum_{j=1}^{R} c_j \prod_{i=1}^{\bar{n}} \binom{y_i}{\bar{S}_{ij}} \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}(t)}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!} \\
&\left(\sum_{i=1}^{\hat{n}-1} \frac{1}{N}(-\theta_l(t)\theta_i(t)) \left(\mathbb{1}(\bar{v}_j = 0)(\frac{\hat{v}_{ij}}{\theta_i(t)} - \frac{\hat{v}_{\hat{n}j}}{\theta_{\hat{n}}(t)}) - \mathbb{1}(\bar{v}_j \neq 0)(\frac{\hat{S}_{ij}}{\theta_i(t)} - \frac{\hat{S}_{\hat{n}j}}{\theta_{\hat{n}}(t)}) \right) \right. \\
&\left. + \frac{1}{N}(\theta_l(t)) \left(\mathbb{1}(\bar{v}_j = 0)(\frac{\hat{v}_{lj}}{\theta_l(t)} - \frac{\hat{v}_{\hat{n}j}}{\theta_{\hat{n}}(t)}) - \mathbb{1}(\bar{v}_j \neq 0)(\frac{\hat{S}_{lj}}{\theta_l(t)} - \frac{\hat{S}_{\hat{n}j}}{\theta_{\hat{n}}(t)}) \right) \right) \\
&= \sum_{j=1}^{R} c_j \prod_{i=1}^{\bar{n}} \binom{y_i}{\bar{S}_{ij}} \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}(t)}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!} \\
&\left(\mathbb{1}(\bar{v}_j = 0) \left(\sum_{i=1}^{\hat{n}-1} \frac{1}{N}(-\theta_l(t))(\hat{v}_{ij}) + \frac{1}{N}\hat{v}_{lj} + \sum_{i=1}^{\hat{n}-1}(\theta_i(t) - 1)\frac{1}{N}(\theta_l(t))(\frac{\hat{v}_{nj}}{\theta_{\hat{n}}(t)}) \right) \right. \\
&\left. - \mathbb{1}(\bar{v}_j \neq 0) \left(\sum_{i=1}^{\hat{n}-1} \frac{1}{N}(-\theta_l(t))(\hat{S}_{ij}) + \frac{1}{N}\hat{S}_{lj} + \sum_{i=1}^{\hat{n}-1}(\theta_i(t) - 1)\frac{1}{N}(\theta_l(t))(\frac{\hat{S}_{nj}}{\theta_{\hat{n}}(t)}) \right) \right) \\
&= \sum_{j=1}^{R} c_j \prod_{i=1}^{\bar{n}} \binom{y_i}{\bar{S}_{ij}} \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}(t)}{\hat{S}_{ij}!} \right) \frac{(N-1)!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!} \\
&\left(\mathbb{1}(\bar{v}_j = 0) \left(\sum_{i=1}^{\hat{n}} (-\theta_l(t))(\hat{v}_{ij}) + \hat{v}_{lj} \right) - \mathbb{1}(\bar{v}_j \neq 0) \left(\sum_{i=1}^{\hat{n}} (-\theta_l(t))(\hat{S}_{ij}) + \hat{S}_{lj} \right) \right).
\end{aligned}$$

For the closed-loop CRN problem considered in the experiments, we get the following drift:

$$\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t} = -c_{12}\,\mathbb{1}(u(t) = 0)\theta_1(t) + c_{21}\,\mathbb{1}(u(t) = 1)\theta_2(t) + c_{24}\theta_1(t)\theta_2(t) + c_{13}\theta_1(t)\theta_1(t) - c_{13}\theta_1(t).$$

We only need to compute the drift for θ_1 as the parameters always need to satisfiy $\theta_1 + \theta_2 = 1$

The filter distribution updates, whenever the observed states jump. If the change vector $\bar{v}$ is seen, the exact filtering distribution is given by:

$$\pi_{t+}(x) = \frac{\sum_{j=1}^{R} \mathbb{1}(\bar{v} = \bar{v}_j) h_j(\hat{x} - \hat{v}_j, \bar{x}(t-)) \pi_{t-}(\hat{x} - \hat{v}_j, \bar{x}(t-))}{\sum_{j=1}^{R} \mathbb{1}(\bar{v} = \bar{v}_j)\, \mathsf{E}\left[h_j(\hat{X}, \bar{X}(t-)) \right]},$$

where the expectation in the denominator is w.r.t. the filter distribution before the jump. We want to approximate the posterior using moment matching. Since we work with the multinomial distribution, we only need to match the first moment M. Here we give the equation for the l-th element of the first moment after the jump $M_{l,t+}$:

$$M_{l,t+} = \frac{\sum_x \hat{x}_l \sum_{j=1}^{R} \mathbb{1}(\bar{v} = \bar{v}_j) h_j(\hat{x} - \hat{v}_j, \bar{x}(t-)) \pi_{t-}(\hat{x} - \hat{v}_j, \bar{x}(t-))}{\sum_{j=1}^{R} \mathbb{1}(\bar{v} = \bar{v}_j)\, \mathsf{E}\left[h_j(\hat{X}, \bar{X}(t-)) \right]} = \frac{\sum_{j=1}^{R} \mathbb{1}(\bar{v} = \bar{v}_j)\, \mathsf{E}\left[(\hat{X}_l + \hat{v}_{lj}) h_j(\hat{X}, \bar{X}(t-)) \right]}{\sum_{j=1}^{R} \mathbb{1}(\bar{v} = \bar{v}_j)\, \mathsf{E}\left[h_j(\hat{X}, \bar{X}(t-)) \right]}.$$

We assume that the filter distribution before the jump belongs to the multinomial family. The expectations are given by:

$$\mathsf{E}\left[h_j(\hat{X}, \bar{X}(t-)) \right] = c_j \prod_{i=1}^{\bar{n}} \binom{y_i(t-)}{\bar{S}_{ij}} \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!},$$

$$\mathsf{E}\Big[(\hat{X}_l + \hat{v}_{lj}) h_j(\hat{X}, \bar{X}(t-))\Big] = c_j \prod_{i=1}^{\bar{n}} \binom{y_i(t-)}{\bar{S}_{ij}} \prod_{i=1}^{\hat{n}} \left(\frac{\theta_i^{\hat{S}_{ij}}}{\hat{S}_{ij}!} \right) \frac{N!}{(N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij})!} \left(\hat{S}_{lj} + \hat{v}_{lj} + \theta_l (N - \sum_{i=1}^{\hat{n}} \hat{S}_{ij}) \right).$$

With this we can compute both the drift and the jump updates in closed-form.

2 DYNAMIC PROGRAMMING

2.1 Dynamic Programming for Continuous-Time MDPs with Discrete State and Action Spaces

We define the value function of the underlying MDP for a state $x \in \mathcal{X}$ as the expected cumulative reward under the optimal control, i.e.,

$$V(x) := \max_{u_{[t,\infty)}} \mathsf{E}\left[\int_t^\infty \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s)) \, \mathrm{d}s \,\middle|\, X(t) = x\right],$$

where we assume that the admissible control $u(t)$ can depend on the state $X(t)$.

By applying the principle of optimality, we can rewrite the value function as:

$$\begin{aligned} V(x) &= \max_{u_{[t,\infty)}} \mathsf{E}\left[\int_t^{t+h} \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s)) \, \mathrm{d}s + \int_{t+h}^\infty \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s)) \, \mathrm{d}s \,\middle|\, X(t) = x\right] \\ &= \max_{u_{[t,t+h)}} \mathsf{E}\left[\int_t^{t+h} \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s)) \, \mathrm{d}s + e^{-\frac{h}{\tau}} V(x(t+h)) \,\middle|\, X(t) = x\right]. \end{aligned}$$

The dynamics of the CTMC are defined through the rate function $\Lambda(x, x', u, t)$:

$$\mathrm{P}(X(t+h) = x' \mid X(t) = x, u(t) = u) = \begin{cases} \Lambda(x, x', u, t)h + o(h) & \text{if } x' \neq x \\ 1 - \sum_{x' \neq x} [\Lambda(x, x', u, t)h + o(h)] & \text{if } x' = x \end{cases}.$$

With these we can compute the expectation of $V(x(t+h))$ and therefore reformulate $V(x)$:

$$\begin{aligned} V(x) = \max_{u_{[t,t+h)}} & \mathsf{E}\left[\int_t^{t+h} \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s)) \, \mathrm{d}s \,\middle|\, X(t) = x\right] \\ &+ e^{-\frac{h}{\tau}} \left(V(x) + \sum_{x' \neq x} [\Lambda(x, x', u, t)h + o(h)] \, (V(x') - V(x))\right). \end{aligned}$$

By bringing the $V(x)$ terms to the left hand side and dividing by h we get

$$\begin{aligned} V(x)(\frac{1 - e^{-\frac{h}{\tau}}}{h}) = \max_{u_{[t,t+h)}} & \frac{1}{h} \mathsf{E}\left[\int_t^{t+h} \frac{1}{\tau} e^{-\frac{s-t}{\tau}} R(X(s), u(s)) \, \mathrm{d}s \,\middle|\, X(t) = x\right] \\ &+ e^{-\frac{h}{\tau}} \left(\sum_{x' \neq x} \left[\Lambda(x, x', u, t) + \frac{o(h)}{h}\right] (V(x') - V(x))\right). \end{aligned}$$

Taking the limit $\lim_{h \to 0}$ we find the optimality conditions as

$$\frac{1}{\tau} V(x) = \max_u \frac{1}{\tau} R(x, u) + \sum_{x' \neq x} \Lambda(x, x', u, t)(V(x') - V(x)),$$

where we define the state action value function as

$$Q(x, u) = R(x, u) + \tau \sum_{x' \neq x} \Lambda(x, x', u, t)(V(x') - V(x)).$$

In the following steps, we show how we can reformulate the value function as a contraction mapping.

$$V(x) = \max_u R(x, u) + \tau \sum_{x' \neq x} \Lambda(x, x', u, t)(V(x') - V(x)).$$

$$V(x)\left(1+\tau\sum_{x'\neq x}\Lambda(x,x',u,t)\right)=\max_u R(x,u)+\tau\sum_{x'\neq x}\Lambda(x,x',u,t)V(x').$$

$$V(x)=\max_u\frac{R(x,u)}{1+\tau\sum_{x'\neq x}\Lambda(x,x',u,t)}+\tau\sum_{x'\neq x}\frac{\Lambda(x,x',u,t)}{1+\tau\sum_{x'\neq x}\Lambda(x,x',u,t)}V(x').$$

With u^* as the maximizer of the right hand side, we can also write it in the following form:

$$Q(x,u^*)=\frac{R(x,u^*)}{1+\tau\sum_{x'\neq x}\Lambda(x,x',u^*,t)}+\tau\sum_{x'\neq x}\frac{\Lambda(x,x',u^*,t)}{1+\tau\sum_{x'\neq x}\Lambda(x,x',u^*,t)}\max_{u'}Q(x',u').$$

2.2 Q-Learning

We built our approximation of the state action value function on the Q-Learning method by Bradtke and Duff (1994). In their work they extended traditional reinforcement learning methods, originally designed for discrete-time MDPs, to encompass SMDPs. In SMDPs the process dynamics are described by semi-Markov processes, which are a generalization of CTMC. Equivalently, our methodology is easily adaptable to scenarios, where the fully observed problem can be described by a SMDP.

In their method, transitions from state x to state x' are sampled while selecting action u. Subsequently, the state-action value function $Q(x,u)$ is updated using the information from the sampled transition and the associated reward $R(x,u)$. In contrast, in our method, we forego the sampling of transitions and, instead, aggregate over all possible transitions, harnessing complete knowledge of the MDP. This is possible, since the possible transitions for each state-action pair are finite.

We use a fully connected neural network $Q(x,u;\theta)$ to approximate the state-action value function. To enhance stability in the learning process, the utilization of a target network $\hat{Q}(x,u;\theta^-)$ is a viable strategy. This secondary network is updated at a slower rate compared to the original network. Algorithm 1 shows the pseudo-code for this Q-Learning method.

Algorithm 1 Q-Learning without Transition Sampling

```
Initialize state-action value function $Q$ with random weights $\theta$
Initialize target state-action value function $\hat{Q}$ with weights $\theta^- = \theta$
Set target update parameter $\kappa$
for episode = 1, M do
    Sample a batch of states $x$
    Get rates $\Lambda(x,x',u,t)$ for all possible next states $x'$ and all actions $u$
    Compute the target $y(x,u) = R(x,u) + \tau\sum_{x'\neq x}\Lambda(x,x',u,t)(\max_{u'}\hat{Q}(x',u';\theta^-) - \max_{u'}\hat{Q}(x,u';\theta^-))$
    Perform a gradient step on the mean squared error between $Q(x,u;\theta)$ and $y$ with respect to $\theta$
    Update the target network: $\theta' \leftarrow \kappa\theta + (1-\kappa)\theta'$
end for
```

3 EXPERIMENTS

3.1 Additional Information on Experiments

The reward function for the considered experiments in the main section is given by $R(x, u) = -\frac{1}{n} \sum_{i=1}^{n} (\frac{x_i - x_i^*}{l})^2$. For the problems with discrete-time measurements we chose Gaussian noise measurements of the exact states, $\mathcal{N}(y \mid x, \sigma^2 I)$. For the queueing problem we observe measurements of queue 2 and queue 3, while the first queue is unobserved. For the LV problem both species are observed. The parameters for the experiments are given in the tables below. In Fig. 7 we provide sample trajectories for the LV problem with a constant control. By comparing these with the results in the main section, we can see that the our control method effectively combines the dynamics of both actions, leading to trajectories that are closer to the goal state.

Table 1: Parameter of the queueing problem

Parameter	**Value**
number of queues n	3
buffer size N	1000
arrival rate λ_1	10.0
arrival rate λ_2	10.0
service rate μ_1	20.0
service rate μ_2	20.0
service rate μ_3	20.0
reward scale l	100.0
goal state x^*	$[0, 0, 0]^\top$
discount τ	5.0
observation noise σ^2	5.0

Table 2: Parameter of the LV problem

Parameter	**Value**
number of species n	2
c_1	2.5
c_2	0.025
$c_3(u = 0)$	1.25
$c_3(u = 1)$	2.5
reward scale l	20.0
goal state x^*	$[100, 100]^T$
discount τ	5.0
observation noise σ^2	5.0

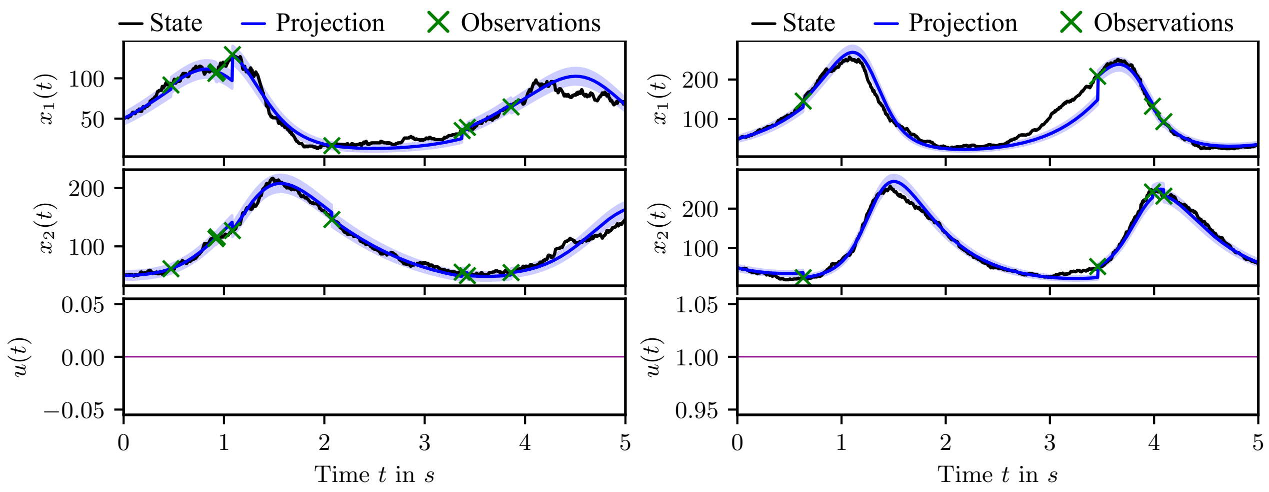

Figure 7: Sample trajectory for the LV problem with constant control.

Table 3: Parameter of the closed-loop CRN problem

Parameter	**Value**
number of species n	4
total species number N	300
c_{12}	0.05
c_{21}	0.05
c_{13}	0.05
c_{31}	0.05
c_{24}	0.05
c_{42}	0.05
c_{34}	0.05
c_{43}	0.05
goal state x^*	$[75, 75, 75, 75]^\top$
reward scale l	1.0
discount τ	5.0

3.2 Projection Filter vs Exact Filter

To be able to evaluate the effect the projection filter has on the control method, we compare it to a policy which employs the QMDP method with respect to the exact filtering distribution. Because exact filtering is only tractable on small state spaces, we create a simpler queueing example based on the experiment in the main section with parameters given in Table 4. For this problem we run 100 sample trajectories for each of the following methods:

- the QMDP method based on the projection filter,
- the QMDP method based on the exact filter,
- and an optimal controller with full knowledge of the state.

For the projection filter we choose again the product binomial distribution as described in Section 5. Fig. 8 shows the kernel density estimates of the cumulative reward for the different polices based on the sample trajectories. Overall the results of all policies are very similar, likely due to the relatively modest scale of the problem at hand. Still we can see that the optimal controller with full knowledge performs best, which is attributable to the fact that the other two methods only have partial observations of the system. The method based on the exact filtering only performs slightly better than the method based on the projection filter. This shows, that the performance

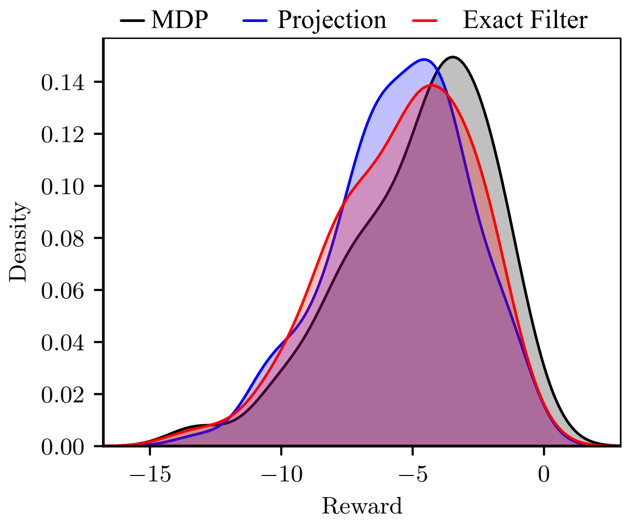

Figure 8: Kernel density estimates of the cumulative reward for different policies using 100 samples.

loss of using the projection method is very small, making it a reasonable choice for larger problems, where the exact filtering method is intractable.

Table 4: Parameter of the queueing problem

Parameter	**Value**
number of queues n	3
buffer size N	5
arrival rate λ_1	1.0
arrival rate λ_2	1.0
service rate μ_1	2.0
service rate μ_2	2.0
service rate μ_3	2.0
reward scale l	1.0
goal state x^*	$[0, 0, 0]^\top$
discount τ	5.0
observation noise σ^2	0.5