Model-free safe reinforcement learning for chemical processes using Gaussian processes

Thomas Savage [*,**] **Dongda Zhang** [**] **Max Mowbray** [**]
Ehecatl Antonio Del Río Chanona [***]

[*] *University of Cambridge, Department of Chemical Engineering and Biotechnology, Philippa Fawcett Drive, Cambridge CB3 0AS*
[**] *University of Manchester, Centre for Process Integration, The Mill, M1 3AL*
[***] *Imperial College London, Centre for Process Systems Engineering, Roderic Hill Building, South Kensington Campus, London, SW7 2AZ*

Abstract: Model-free reinforcement learning has been recently investigated for use in chemical process control. Through the iterative creation of an approximate process model, control actions are able to be explored and optimal policies generated. Typically, this approximate process model has taken the form of a neural network that is continuously updated. However when small quantities of historical data are available, for example in novel processes, neural networks tend to over-fit to data providing poor performance. In this paper Gaussian processes are used as a method of function approximation to describe the action-value function of a non-isothermal semi-batch reactor. Through the use of analytical uncertainty obtained from Gaussian process predictions, trade off between exploration and exploitation is enabled, allowing for efficient generation of effective policies. Importantly Gaussian processes also enable probabilistic constraint violation to be modelled, ensuring safe constraint satisfaction throughout the learning procedure. On application to the in-silico case study, a safe, effective policy was generated utilising only 100 evaluations of process trajectory with no prior knowledge of the process dynamics. A result that would require significantly more trajectory evaluations when compared to a neural network based approach.

Keywords: Batch Processes, Modelling and Identification, Scheduling and Optimization

1. INTRODUCTION

The control and optimisation of nonlinear stochastic processes poses a complex task for existing control schemes. Typically, the approach provided by direct optimal control (DOC) necessitates methods to simultaneously account for process stochasticity and handle constraints. This often requires assumptions regarding the nature of process uncertainty and the manner in which it is propagated. Furthermore, DOC methods are dependent upon the availability of an accurate model of the physics of the underlying process for online computation of optimal control actions. In the case of nonlinear, stochastic processes, such descriptions are often difficult to identify and provide an increase in online computational cost. This is discussed further in Mayne (2015); Forbes et al. (2015). An alternative control solution is provided in the form of model-free reinforcement learning (RL).

1.1 Model-Free Reinforcement Learning in the Chemical Process Industries

The application of model-free RL to chemical process control has been investigated for many years, dating back as far as the early 1990s, as presented by Hoskins and Himmelblau (1992). Since then, the wider field of RL has observed impressive breakthroughs through algorithm development in game-based control benchmarks e.g. Silver et al. (2018). Many of these algorithms have been demonstrated within the context of process control. In Spielberg et al. (2019), the authors present an actor-critic algorithm for control through a number of case studies, including a high purity distillation column. In Lee and Lee (2006), an action-value method is presented in the context of both process control and scheduling. A number of publications have also investigated application of RL to the control of nonlinear stochastic processes as demonstrated by Ma et al. (2019); Lee and Lee (2005); Petsagkourakis et al. (2020b). All of these works assume the availability of an offline process model for the purposes of policy learning. In Hwangbo and Sin (2020), an action-value method is presented for control of a downstream separation in a biopharmaceutical process. Importantly, focus within RL-orientated academic works is increasingly directed towards guarantee of safe process operation and policy learning. This is underpinned by recent works which provide approaches for safe constraint satisfaction e.g. Petsagkourakis et al. (2020a). For further context on the application of RL in the process industries, we direct the reader to a recent review provided by Shin et al. (2019).

1.2 Motivation

Despite the achievements and applications described previously, RL currently observes severe obstacles to reliable implementation as a control solution within the process industries. Mainly, concerns are directed to three areas: satisfaction of operational constraints, process-model mismatch (if offline learning via simulation is used), and epistemic uncertainties in parameterisation of the control policy. Further, conventional approaches to RL leverage the use of artificial neural networks (ANNs) for the purpose of policy or value function parameterisation. The use of ANNs demands the generation of large datasets for learning, and in the case of 'on-policy' learning algorithms, much of this data may only be used for one learning update before being discarded. This provides practically prohibitive cost in learning an optimal policy for a novel process whereby existing historical data is unavailable. Therefore, as previously alluded, initial policy learning is conducted offline and demands simulation of the process via an approximate process model. This reality represents a dualism inherent to RL methods, which is often ignored when demonstrated empirically in case study. Additionally, ANN models are often prone to overfitting when the amount of data available is significantly smaller than the number of parameters, and control performance may be highly sensitive to specific configurations and architectures Zhang et al. (2018).

Gaussian processes (GPs) have been used for a number of years as surrogate models (also known as meta-models) for the control and optimisation of chemical processes, see for example Bradford et al. (2020, 2018). A key advantage of GPs, as statistical models, is that they are able to encode analytical uncertainty as a prediction consists of the posterior probability distribution over possible function values. This provides a mechanism for control to account for process and policy (or value function) parameterisation uncertainties - providing an inherently safer alternative to *vanilla* use of ANN. Consequently, techniques such as Bayesian optimisation have taken advantage of this in order to find global solutions to otherwise expensive optimisation problems Jones et al. (1998). As such, the integration of GPs with the decision-making framework underpinning RL provides an attractive prospect in the scope of sample efficient and safe policy learning. In the following, a method is proposed, which combines GP-based RL and Bayesian optimisation with the concept of constraint-tightening and backoffs. The concept of constraint tightening is translated from the domain of stochastic model predictive control (sMPC) and enables the probabilistic satisfaction of constraints Mehta and Ricardez-Sandoval (2016); Bradford et al. (2020). It is also hypothesised that the method could reduce dependence upon offline simulation provided data from an existing control scheme is available.

The structure of this paper is as follows: Section 2 first introduces Gaussian processes and Q-learning, then the proposed method that integrates these two methods is outlined. Subsequently, the benefits of using GPs as Q-functions are described and the case study is then introduced applying the proposed methodology to the control of a non-isothermal semi-batch reactor. Section 3 outlines the results and discussion of the application of GP Q-learning to the case study and Section 4 provides a subsequent conclusion and directions of future work.

2. METHODOLOGY

2.1 Gaussian Processes

A Gaussian process is defined as *a collection of random variables, any finite number of which have a joint Gaussian distribution* Rasmussen and Williams (2005). Specified by a mean function and a covariance function, Gaussian processes are non-parametric and as a result, concepts inherent in parametric modelling such as over and underfitting data are avoided. The mean function is commonly set to a constant value of 0 as this specifies as limited prior information regarding the underlying function. Subsequently, a Gaussian process is defined as follows:

$$f(x) \sim \mathcal{GP}(0, k(\cdot,\cdot)) \tag{1}$$

where f is the resulting function approximation, and $k(\cdot,\cdot)$ is the covariance function.

The covariance function is used to relate the correlation between neighboring data in input-space, most commonly through a distance metric such as a norm. Often used due to its smooth, differentiable form is the squared exponential kernel:

$$k(\mathbf{x}, \mathbf{x}^*) = \sigma_f^2 \exp\left(-\frac{1}{2l^2}(||\mathbf{x} - \mathbf{x}^*||_2^2)\right) + \sigma_n^2 \delta_{pg} \tag{2}$$

where $\mathbf{x}$ and $\mathbf{x}^*$ are vectors of inputs, l and σ_f are hyper-parameters that effect the length scale of functions that are produced, σ_n defines the level of noise present in the posterior distribution and δ_{pg} is the Kronecker-delta function. If for example data was derived from noise-free computational experiments, this parameter could be manually selected as 0 thus specifying zero posterior uncertainty when $\mathbf{x} = \mathbf{x}^*$ i.e. predicting the value of the function at a known input. These hyper-parameters are optimised by minimising the negative log-likelihood of the Gaussian process along with respective data:

$$-\log p(\mathbf{y}|X) = \frac{1}{2}\mathbf{y}^T(K + \sigma_n^2 I)^{-1}\mathbf{y} + \frac{1}{2}\log|K + \sigma_n^2 I| + \frac{n}{2}\log 2\pi \tag{3}$$

where $\mathbf{y}$ is the set of observed function values, X is the training data matrix, K is the gram-matrix produced by covariance function k and the training data, and n is the dataset size.

The posterior distribution after evaluating the Gaussian process at novel input locations X_* is given by the following multivariate Gaussian distribution.

$$\mathbf{f}_*|X_*, X, \mathbf{f} \sim \mathcal{N}(\mu, \mathbf{\Sigma}) \tag{4}$$

$$\mu = K(X_*, X)K(X, X)^{-1}\mathbf{f} \tag{5}$$

$$\mathbf{\Sigma} = K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*)) \tag{6}$$

where $K(X_*, X)$ is the Gram matrix between the training, and evaluation locations, with $K(X, X_*)$ being its transpose. For further details refer to Rasmussen and Williams (2006).

2.2 Action-value Learning

Q-Learning is a reinforcement learning technique developed by Watkins Watkins (1989). This method takes advantage of a learned action-value function in order to determine an optimal control policy Sutton and Barto (2018). A Q-function is created that takes both states and controls, and outputs an expected future reward. By optimising over possible control actions at each time-step, the future reward given the current state can be maximized.

Maintaining consistent notation for remaining sections, the Q-function temporal difference update is performed as follows:

$$Q(\mathbf{x}_t, \mathbf{u}_t) = \mathbb{E}_\pi \left[\sum_{t'=t}^{T-1} R_{t'+1} | \mathbf{x} = \mathbf{x}_t, \mathbf{u} = \mathbf{u}_t \right] \tag{7}$$

where Q is a function that describes the total expected future reward given a state and an action. R_{t+1} is the reward given for taking the action $\mathbf{u}_t$ at state $\mathbf{x}_t$ and T is the length of the finite control horizon. γ defines how much future rewards are discounted and may or may not be included depending on the length of the control horizon, and α defines a learning rate to be used when the Q function takes the form of a table (discrete states and controls). Once the actual reward of the system has been noted following enactment of the controls chosen, the Q function is updated allowing for a better subsequent determination of rewards give states.

For a discrete-time process with a set of discrete control actions, a Q-function is created at each discrete time-step. Note that for the initial creation of the Q-functions, the state and control space is originally sampled using an efficient space-filling regime (e.g. Latin Hypercube Sampling (LHS)) in order to create a search space over which the future reward can be optimised. For a process with N discrete time steps, the Q-learning algorithm is as follows:

Algorithm 1. N-step Q-learning

Initialisation: Sample the state-action space and construct n Q-functions Q_i for $i = 1, ..., n$, given iterations and initial state $\mathbf{x_0}$.
while *iterations not reached* **do**
  for *n in N-steps* **do**
    Provide Q_n with $\mathbf{x}_n$
    $\mathbf{u}_{opt} = \arg\max_{\mathbf{u}} Q_n(\mathbf{x}_n, \mathbf{u})$
    Take control action $\mathbf{u}_{opt}$ and observe $\mathbf{x}_{n+1},\ R_{n+1}$
  end
  Updates Q-functions:
  for *n in N-steps* **do**
    $Q_n(\mathbf{x}_n, \mathbf{u}_{opt}) = R_n + Q_{n+1}$
  end
end
Output: n Q-functions that provide an optimal control policy when optimised control actions are taken at each step.

2.3 GP Q-Learning

Q-Learning and its many variants have found much success in recent years through the use of deep neural networks (DNNs) to approximate the Q-function. However with this comes a number of challenges, namely the amount of data required to properly train a DNN, address overfitting, as well as the time consuming hyperparameter selection and neural architecture searches that are vital to ensure the success of this approach. The issue also becomes prevalent when considering that the majority of RL case studies and successes derive from applications with rapid evaluation of states and actions, for example a computer playing itself at chess or a video game, or even robotics. It is much more time consuming to gain this vital data within the context of a chemical process. For example chemical or biochemical reactions may take hours or days to complete. Therefore, for RL technologies to be effectively utilised within the process industry, a more sample efficient approach must be taken with regards to function approximation. Thus motivates the use of Gaussian processes as an internal model, enabling model-free RL techniques to be taken advantage of.

By instead approximating the Q-function at each timestep by a separate Gaussian process a number of advantages can be derived.

Automatic exploration/exploitation.
Traditional reinforcement learning approaches to balancing the exploration of the search space and the exploitation of existing known control actions can be heuristical such as the $\epsilon-$greedy approach of taking a random action with a certain decreasing probability. As Gaussian processes provide analytical uncertainty into a prediction, it naturally follows that Bayesian optimisation techniques can be taken advantage of. By optimising the mean in addition to the standard deviation, areas of high uncertainty are automatically explored. In the case of maximisation of the Q-function with respect to control actions the Upper Confidence Bound (UCB) is defined as:

$$Q_n^{\mathrm{UCB}}(\mathbf{x_n}, \mathbf{u}) = \mathbb{E}\left[Q_n(\mathbf{x_n}, \mathbf{u})\right] + \beta\sigma\left[Q_n(\mathbf{x_n}, \mathbf{u})\right] \tag{8}$$

where $\mathbb{E}[Q_n(\mathbf{x_n}, \mathbf{u})]$ is the mean of the GP posterior distribution, σ denotes the standard deviation of this posterior, and β is a scaling factor. By optimising this UCB Q-function at each timestep, automatic exploration and exploitation is enabled allowing for optimal policies to be found efficiently. This is also illustrated in Fig. 1 where the underlying function is shown in red, an approximating Gaussian process mean (black) and variance (grey) is shown on the same axis. Below, the composite UCB function is shown illustrating how the search space is modified to encourage exploration to regions with high variance.

More sample efficient.
As previously mentioned, it can take hours or days for some chemical processes to reach completion. For an existing process a historical dataset may be able to be used, however for a novel process real world data will initially be sparse. Therefore if model-free RL techniques such as Q-learning are to be used effectively, they will need to learn effectively from initially little data.

As Gaussian processes specify a *smoothing* model class (i.e. predictions weighted by the training data outputs) as opposed to a regression model whereby the model's prediction is calculated through the minimisation of an

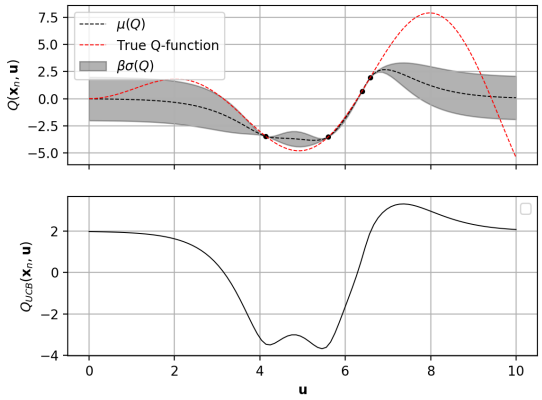

Fig. 1. Example visualisation of the UCB Bayesian optimisation approach to optimising each Q-function.

error metric (as is the case with neural networks), they are less prone to over or underfitting when compared to ANNs.

On the contrary, too much data can cause Gaussian processes to fail due to inefficient scaling of the inversion of the covariance matrix, however in recent years considerable effort has been made to enable Gaussian processes to be efficient for larger datasets, whilst also maintaining efficiency at low data numbers (Josiah Yan , Thang D. Bui , Turner (2017)).

Probabilistic constraint violation (back-off).
By modelling the future constraint violation at each time-step using a distinct Gaussian processes in a similar manner to modelling the expected future reward (Q-function), constraints can be handled with relative ease. The output of this constraint function at each time-step is appended as a penalty to the cost function, as is customary when using evolutionary algorithms, although complete global optimization can be used (Schweidtmann et al., 2020). The augmented Q-function to be optimized at each time-step is therefore a combination of the Q-function itself, the variance as described previously, and the constraint penalty with a backoff term:

$$\max_{\mathbf{u}} \ \mathbb{E}[Q_n(\mathbf{x}_n,\mathbf{u})] + \beta\sigma[Q_n(\mathbf{x_n},\mathbf{u})] - \omega\sum_{i=1}^{i}\max(\mathbb{E}[\mathrm{GP}^i_{con}(\mathbf{x_n},\mathbf{u})] + \sigma[\mathrm{GP}^i_{con}(\mathbf{x_n},\mathbf{u})], 0) \tag{9}$$

Where GP^i_{con} denotes the GP associated with the prediction of constraint i and ω is the weight of this penalty. As there is also an associated uncertainty with these constraint violation predictions, a probabilistic back-off term may be appended to the expected function value proportional to the standard deviation of the posterior predictive distribution.

In this way, the probability of a constraint violation can be specified, ensuring safe exploration during the model-free RL procedure.

2.4 Case Study

The case study used to evaluate the viability of Q-Learning through Gaussian process function approximation is the control of a semi-batch non-isothermal reactor with multiple reactions. Specifically, series reactions of the form:

$$2\mathrm{A} \xrightarrow{k_{1,A}} \mathrm{B} \xrightarrow{k_{2,B}} 3\mathrm{C} \tag{10}$$

The set of five process states $\mathbf{x}$ consists of $[\mathrm{C_A}, \mathrm{C_B}, \mathrm{C_C}, \mathrm{T}, \mathrm{V}]$ with the controlled variables $\mathbf{u}$ consisting of the heat exchanger temperature $\mathrm{T_a}$ and inflow rate of A, F. The dynamics of the reactor are defined as:

$$\frac{\mathrm{dC_A}}{\mathrm{dt}} = -k_{1A}\cdot\mathrm{C_A} + (\mathrm{C_{A0}} - \mathrm{C_A})\cdot\frac{F}{V} \tag{11}$$

$$\frac{\mathrm{dC_B}}{\mathrm{dt}} = k_{1A}\cdot\mathrm{C_A}/2 - k_{2B}\cdot\mathrm{C_B} - \mathrm{C_B}\cdot\frac{F}{V} \tag{12}$$

$$\frac{\mathrm{dC_C}}{\mathrm{dt}} = 3k_{2B}\cdot\mathrm{C_B} - \mathrm{C_C}\cdot\frac{F}{V} \tag{13}$$

$$\frac{\mathrm{dV}}{\mathrm{dt}} = \mathrm{F} \tag{14}$$

$$\frac{\mathrm{dT}}{\mathrm{dt}} = \frac{\mathrm{UA(T_a - T)} - F_{A0}\mathrm{C}_{P_A}(\mathrm{T}-\mathrm{T_0}) + \ldots}{[\mathrm{C_A C_{P_A}} + \mathrm{C_B C_{P_B}} + \mathrm{C_C C_{P_C}}]\mathrm{V} + N_{H_2SO_4}\mathrm{C}_{P_{H_2SO_4}}} \frac{[(\Delta H_{R1A})(-k_{1A}\cdot\mathrm{C_A}) + (\Delta H_{R2B})(-k_{2B}\cdot\mathrm{C_B})]V}{[\mathrm{C_A C_{P_A}} + \mathrm{C_B C_{P_B}} + \mathrm{C_C C_{P_C}}]\mathrm{V} + N_{H_2SO_4}\mathrm{C}_{P_{H_2SO_4}}} \tag{15}$$

where both reaction rate constants are defined respectively as:

$$k_{1A} = A_1\cdot\exp\left[E_{1A}\cdot\left(\frac{1}{\mathrm{T}_{r_1}} - \frac{1}{\mathrm{T}}\right)\right] \tag{16}$$

$$k_{2B} = A_2\cdot\exp\left[E_{2B}\cdot\left(\frac{1}{\mathrm{T}_{r_2}} - \frac{1}{\mathrm{T}}\right)\right] \tag{17}$$

Parameters that are neither states nor controls as previously defined in Equations 11 to 17, have fixed values which can be found in Fogler (2006).

Scaled Gaussian noise is added to the dynamics of each state to induce stochasticity within the case study, providing a more representational real world problem.

Dynamics are evaluated over the course of 4 hours using a Runge-Kutte 4th order integration scheme, at which point the final amount of product $\mathrm{C_C}\cdot V$ is evaluated. Control actions are assumed to be constant inputs to the system that can be changed once per 24 minutes, resulting in a total of 10 control actions for each batch. Each input is bounded, and the overall optimal control problem can be defined as follows.

$$\max_{\mathbf{u}} \quad \mathrm{C_C}(t_f)\cdot V(t_f) \tag{18}$$

$$\text{s.t.} \quad \dot{\mathbf{x}} = f(\mathbf{x}(t),\mathbf{u}(t),t) \tag{19}$$

$$\mathbf{x}_0 = [1,0,0,290,100] \tag{20}$$

$$T(t) < 420\ K \tag{21}$$

$$V(t) < 800\ L \tag{22}$$

$$0\ Lh^{-1} \le F(t) \le 270\ Lh^{-1} \tag{23}$$

$$298K \le T_a(t) \le 500K \tag{24}$$

where $\mathbf{x}$ is the set of process states, $\mathbf{u}$ is the set of controlled variables. Two additional constraints concern the overall

temperature of the reactor at any one time being limited to 420 K, and the total volume of the reactor being limited to 800 L.

The optimal control problem is solved through the creation and subsequent optimisation of a separate Q-function at each time-step. This Q-function encodes information about the future reward given the current state and control action. By optimising over the control actions, the future reward (in this case production of C) is maximised. Once a control action is chosen, the dynamics are integrated over that time interval and the next Q-function is provided the resulting state and subsequently optimised with respect to its control action. Once a complete trajectory has been undertaken, all Q-functions are updated with the previous state at the specific time-step, control action at each time-step, and resulting reward (immediate and discounted future).

Constraints are handled in a similar fashion, with a separate Gaussian process at each time-step predicting possible future constraint violation given the current state and a possible control action. However the 'reward' becomes a penalty proportional to the violation of each constraint. By appending this penalty to the cost function of the optimisation problem, control actions can be chosen that minimise the probability of violating a constraint both immediately and in the future.

Gaussian process based Q-functions were implemented in Python 3.7.4 using the GPy library (GPy (since 2012)) and their optimisation was performed using a BFGS (Liu and Nocedal (1989)) solver within the SciPy scientific computing library.

3. RESULTS AND DISCUSSION

The semi-batch non-isothermal reactor case study was first sampled 20 times using control actions generated via a Latin hypercube sample in order to gain a representative, distributed, set of initial inputs. In an industrial setting historical process data may be used to initialise the GPs. These control actions were used to gain initial values over the constraint and objective function (future reward) space which were subsequently used along with the generated inputs to construct the initial GPs. Following this, the model-free Q-learning algorithm was applied and each GP was updated with new values after each iteration. The algorithm was left to run for 100 iterations, a real-life time of around 16 days assuming a single reactor in operation and no maintenance or cleaning between batches. Computationally the complete routine took 35 minutes on a Macbook Pro with a 1.4 GHz i5 8^{th} generation processor.

The average results over the final 50 iterations are shown below with the grey area representing one standard deviation. The first 5 axis show the 5 process states, dotted red-lines indicate the two process constraints shown above, and the final 3 axis show the two respective control actions, and total production of the product C in kmol.

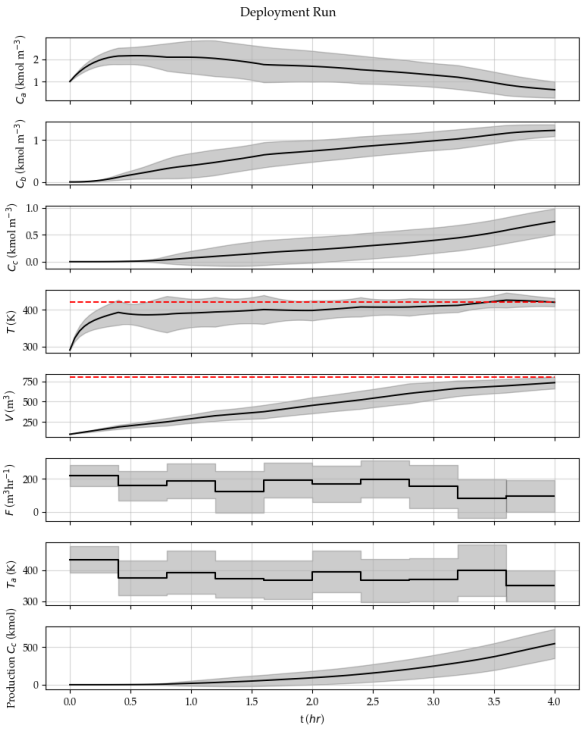

Fig. 2. Average states, control actions and production of the final 50 iterations of Q-learning, each iteration consisting of a distinct 4 hour process trajectory. Grey bars represent one standard deviation.

In Fig. 2, it can be seen that effective control policies for both F and T_a were identified, with the F control policy slightly more constrained than T_a. This may be as a result of the more rapid coupling between the temperature control and the temperature of the reactor. When considering the controller has to adjust to account for this temperature constraint across the majority of the process trajectory, a broader range of control actions are sought in order to quickly satisfy this constraint. Importantly the two process constraints on volume and temperature are both largely respected throughout the trajectories shown within one standard deviation. Near the end of the process trajectory the average temperature of the reactor rises slightly above the value of the constraint. However, it is anticipated as is the case with reinforcement learning in general that more considered hyper-parameter tuning would negate this small infraction. The effect of probabilistic back-offs, facilitated by the use of GPs is also made clear as both constraints are respected within one standard-deviation. Subsequently, the trajectories generated by the model-free GP RL algorithm are shown to be safe, a key aspect of any algorithm that is to be used within the control of industrial processes.

4. CONCLUSIONS AND FUTURE WORK

To conclude, in this paper model-free action-value reinforcement learning was introduced and outlined in the context of processes with discrete control-actions in time. Gaussian processes were then introduced as alternatives to traditional neural networks for the creation of action-

value functions. With benefits namely including analytical uncertainty, utilised in Bayesian optimisation to balance the exploration-exploitation trade-off, as well as increased efficiency for low volumes of data. Gaussian processes also have the advantage in not over or underfitting providing an accurate action-value function for all magnitudes of data available. Process constraints were represented again using Gaussian processes, with the analytical uncertainty providing a 'back-off' to the constraint penalty term ensuring safe-exploration.

A non-isothermal semi-batch reactor was used as a case-study and a control policy generated using 100 iterations of the model-free GP RL algorithm described. Constraints were seen to be respected within one standard deviation, a value specified as the backoff, and the production of chemical product was maximised.

Future work may wish to consider the direct comparison between Gaussian processes and a neural-network based approach. Improvements could also be investigated regarding the considered removal of past datapoints that contribute to older, sub-optimal states and actions as well as sensitivity to initialisation of GPs. Analogous to replay-memory in neural-network based RL, a considered approach into which datapoints are provided to each GP may result in further improved optimal policies. It may also be possible to parallelise the training of the separate GPs allowing for faster optimisation times. However, when considering the time to complete a single process trajectory in real-time compared to GP training and optimisation, further development in reducing computational time may not be required and attention should be placed elsewhere (e.g. identifying optimal policies).

REFERENCES

Bradford, E., Imsland, L., Zhang, D., and del Rio Chanona, E.A. (2020). Stochastic data-driven model predictive control using gaussian processes. *Computers & Chemical Engineering*, 139, 106844.

Bradford, E., Schweidtmann, A.M., Zhang, D., Jing, K., and del Rio-Chanona, E.A. (2018). Dynamic modeling and optimization of sustainable algal production with uncertainty using multivariate gaussian processes. *Computers & Chemical Engineering*, 118, 143–158.

Fogler, H.S. (2006). *Elements of chemical reaction engineering*. Prentice Hall Professional.

Forbes, M.G., Patwardhan, R.S., Hamadah, H., and Gopaluni, R.B. (2015). Model predictive control in industry: Challenges and opportunities. *IFAC-PapersOnLine*, 48(8), 531–538.

GPy (since 2012). GPy: A gaussian process framework in python.

Hoskins, J. and Himmelblau, D. (1992). Process control via artificial neural networks and reinforcement learning. *Computers & chemical engineering*, 16(4), 241–251.

Hwangbo, S. and Sin, G. (2020). Design of control framework based on deep reinforcement learning and monte-carlo sampling in downstream separation. *Computers & Chemical Engineering*, 106910.

Jones, D.R., Schonlau, M., and Welch, W.J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4), 455–492.

Josiah Yan , Thang D. Bui , Turner, R.E. (2017). A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation. *Journal of Machine Learning Research 18*, 18, 1–72.

Lee, J.H. and Lee, J.M. (2006). Approximate dynamic programming based approach to process control and scheduling. *Computers & chemical engineering*, 30(10-12), 1603–1618.

Lee, J.M. and Lee, J.H. (2005). Approximate dynamic programming-based approaches for input–output data-driven control of nonlinear processes. *Automatica*, 41(7), 1281–1288.

Liu, D.C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3), 503–528.

Ma, Y., Zhu, W., Benton, M.G., and Romagnoli, J. (2019). Continuous control of a polymerization system with deep reinforcement learning. *Journal of Process Control*, 75, 40–47.

Mayne, D. (2015). Robust and stochastic mpc: Are we going in the right direction? *IFAC-PapersOnLine*, 48(23), 1–8.

Mehta, S. and Ricardez-Sandoval, L.A. (2016). Integration of design and control of dynamic systems under uncertainty: A new back-off approach. *Industrial & Engineering Chemistry Research*, 55(2), 485–498.

Petsagkourakis, P., Sandoval, I.O., Bradford, E., Galvanin, F., Zhang, D., and del Rio-Chanona, E.A. (2020a). Chance constrained policy optimization for process control and optimization. *arXiv preprint arXiv:2008.00030*.

Petsagkourakis, P., Sandoval, I.O., Bradford, E., Zhang, D., and del Rio-Chanona, E.A. (2020b). Reinforcement learning for batch bioprocess optimization. *Computers & Chemical Engineering*, 133, 106649.

Rasmussen, C. and Williams, K. (2006). *Gaussian processes for machine learning*. doi: 10.1142/S0129065704001899.

Rasmussen, C.E. and Williams, C.K.I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Schweidtmann, A.M., Bongartz, D., Grothe, D., Kerkenhoff, T., Lin, X., Najman, J., and Mitsos, A. (2020). Global optimization of gaussian processes.

Shin, J., Badgwell, T.A., Liu, K.H., and Lee, J.H. (2019). Reinforcement learning–overview of recent progress and implications for process control. *Computers & Chemical Engineering*, 127, 282–294.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419), 1140–1144.

Spielberg, S., Tulsyan, A., Lawrence, N.P., Loewen, P.D., and Bhushan Gopaluni, R. (2019). Toward self-driving processes: A deep reinforcement learning approach to control. *AIChE Journal*, 65(10), e16689.

Sutton, R. and Barto, A. (2018). *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press.

Watkins, C.J.C.H. (1989). Learning from delayed rewards.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. (2018). A study on overfitting in deep reinforcement learning.