Deep Reinforcement Learning for process control and optimization

Bruno Didier Olivier Capron

Escola de Química da UFRJ

bruno@eq.ufrj.br

PSE-BR

Rio de Janeiro

21/05/2019

Introduction

- Limitations of model-based control techniques for nonlinear processes (Xi et al., 2013)
  - Rigorous models of nonlinear processes are hard and time consuming to obtain
  - Even when available, large computational effort is needed for solving the nonlinear process control optimization problem
  - Models deteriorate in time so that they need to be reevaluated or adaptive mechanisms need to be implemented

- Linear models are still mostly used in practice

Introduction

- Successful applications of the data-based Reinforcement Learning techniques

Process Control?

- Automation and data acquisition systems have grown in use in the process industry over the last decades
- The number of publications of RL-based process control applications is blooming (Hoskins and Himmelblau, 1992; Anderson et al., 1997; Martinez, 2000; Syafiie et al., 2008; Li et al., 2011; Mustafa and Wilson, 2012; Shah and Gopal, 2016; Ramanathan et al., 2017; Hernández del Olmo et al., 2017; Pandian and Noel, 2018; Ma et al., 2019)

Introduction

- Interesting characteristics of RL-based control:
  - No need for a rigorous model of the process
  - The controller may be pre-trained from already implemented controllers (PID, MPC,…) or simulations based on simple models
  - RL-based control techniques are naturally adaptive as the learning may continue with the integration of new process data
  - By construction, the computational effort is very low compared to classical model-based techniques

- In this work, a deep reinforcement learning algorithm for continuous control (Lillicrap et al., 2016) is implemented for the control and optimization of a Van De Vusse reactor

Reinforcement Learning in the Machine Learning context

Machine Learning

Supervised Learning

Learning of a function from a data set of inputs and outputs.

Classification

Regression

Unsupervised learning

Learning of a function from a data set. (The data are not classified nor categorized)

Clustering

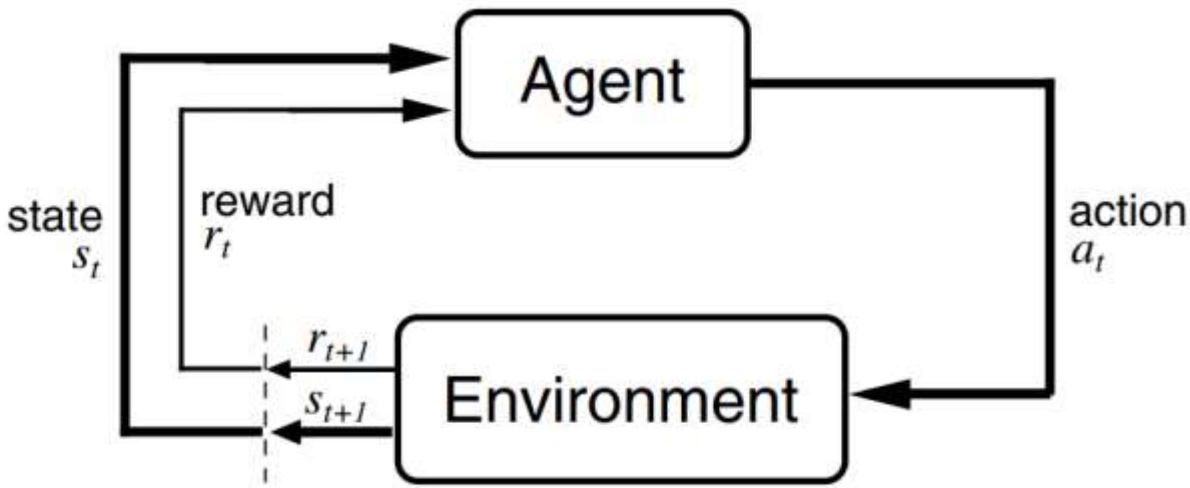
Non clustering

Reinforcement Learning

Learns a specific task by direct interaction with the environment from reinforcement signals received from the environment.

Reinforcement Learning Problem

- An agent (controller) learns a specific task (process control) by direct interaction with the environment (process)



(Sutton and Barto, 1998)

- Objective: learning a *policy* that maximizes the prevision of acumulated rewards. In the deterministic case:

$$\pi(s_t) = a_t$$

Return

- Given a sequence of steps (episode) represented by:

$$s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, s_{t+2}, a_{t+2}, r_{t+2}, s_{t+3} \ldots$$

- The return R_t is the total discounted reward from time-step t onwards

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots = \sum_{k=0}^{+\infty} \gamma^k r_{t+k}, \gamma \in [0,1]$$

Action-value function

- The action-value function is the expected return given a certain state s_t and a given action a_t following policy π

$$Q_\pi(\mathrm{s}, a) = \mathrm{E}_\pi \left\{ \sum_{k=1}^{+\infty} \gamma^{k-1} r_k \mid S_t = \mathrm{s}, A_t = a \right\}$$

- Objective: Find the optimal action-value function Q*(s,a) that is the maximum action-value function among all possible policies:

$$Q^*(\mathrm{s}, \mathrm{a}) = \max_\pi \mathrm{Q}_\pi(\mathrm{s}, \mathrm{a})$$

Best policy

- The best policy may be obtained by evaluating the action that maximizes Q*(s,a):

$$\pi(\mathrm{s}) = \arg\max_{a \in A} Q^*(\mathrm{s},\mathrm{a})$$

- Problem easily solved for discrete state and action spaces
- 2 options for continuous problems (as for process control):
  - Discretization of the state and action spaces but the problem is affected by the "curse of dimensionality"
  - No discretization is done but the above optimization problem becomes a NonLinear Programming optimizaton problem (hard and time consuming to solve)

Action-Value function approximation

- To avoid the "curse of dimensionality", continuous action and state spaces are considered and the action-value function is approximated by a parameter-dependent function:

$$Q(\mathrm{s}, \mathrm{a} \mid \theta^{Q})$$

- Possible approximators:
  - Linear combination of radial basis functions
  - **Neural networks**
  - …

Deep Reinforcement Learning

Actor-Critic algorithm

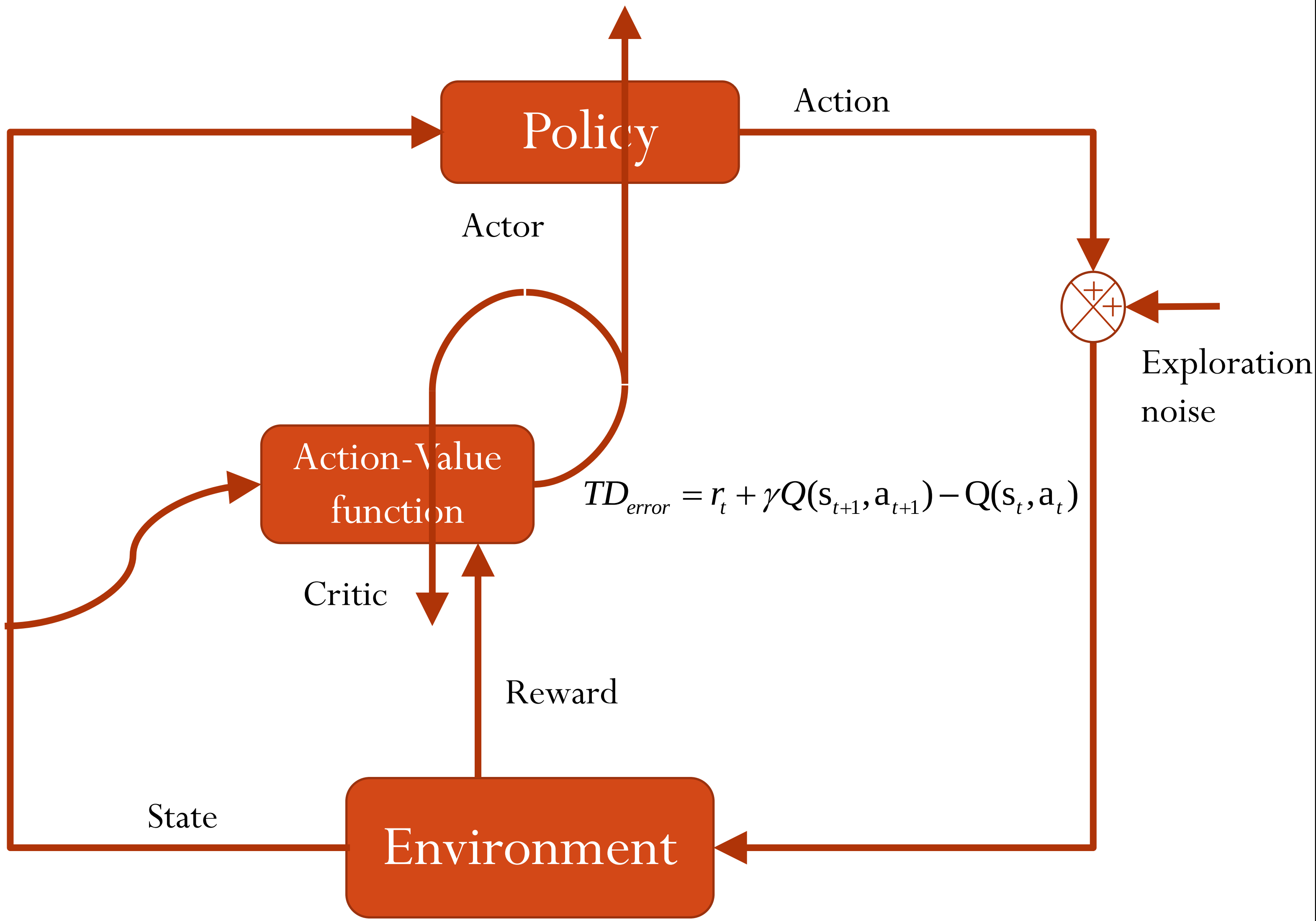
$$\cancel{\pi(s) = \arg\max_{a \in A} Q^*(s,a)}$$

- Allows the use of continuous state and action spaces without solving a complex NLP optimization problem
- As the action-value function, the policy is explicitly represented by a parameter-dependent function (neural network here)

$$\pi(s \mid \theta_\pi)$$

- An iterative algorithm for training:
  - The actor is responsible for improving the policy
  - The critic is responsible for evaluating the action-value function

Actor-Critic algorithm



Adapted from (Sutton and Barto, 1998)

Deep Deterministic Policy Gradient (DDPG) algorithm (Lillicrap et al., 2016)

- Both the policy and the action-value function are represented by neural networks:
  - Action-value function:

$$Q(\mathrm{s}, \mathrm{a} \mid \theta^{Q})$$

  - Policy

$$\pi(\mathrm{s} \mid \theta_{\pi})$$

- At each time step, the networks are updated with samples from a replay buffer to minimize correlations between samples

Deep Deterministic Policy Gradient (DDPG) algorithm

- At each time step, select action according to current policy and exploration noise:

$$a_t = \pi(s_t \mid \theta^{\pi}) + \eta_t$$

- Implement action at and observe immediate reward r_t and new state s_{t+1}
- Store this transition defined by (s_t, a_t, r_t, s_{t+1})
- Sample a random batch of N transitions (s_i, a_i, r_i, s_{i+1}) from the replay buffer
- Set

$$y_i = r_i + \gamma Q(s_{i+1}, \pi(s_{i+1} \mid \theta^{\pi}))$$

Deep Deterministic Policy Gradient (DDPG) algorithm

- Update the critic network aiming at minimizing the loss defined by:

$$L = \frac{1}{N}\sum_i \left(y_i - Q(\mathrm{s}_i, \mathrm{a}_i \mid \theta^Q) \right)^2$$

- Update the actor network (policy) in the direction of greater cumulative reward using the sampled policy gradient:

$$\frac{1}{N}\sum_i \nabla_a Q(\mathrm{s}, \mathrm{a} \mid \theta^Q)\big|_{s=s_i, a=\pi(\mathrm{s}_i)} \nabla_{\theta^\pi} \pi(\mathrm{s} \mid \theta^\pi)\big|_{s_i}$$

CSTR with Van de Vusse reaction

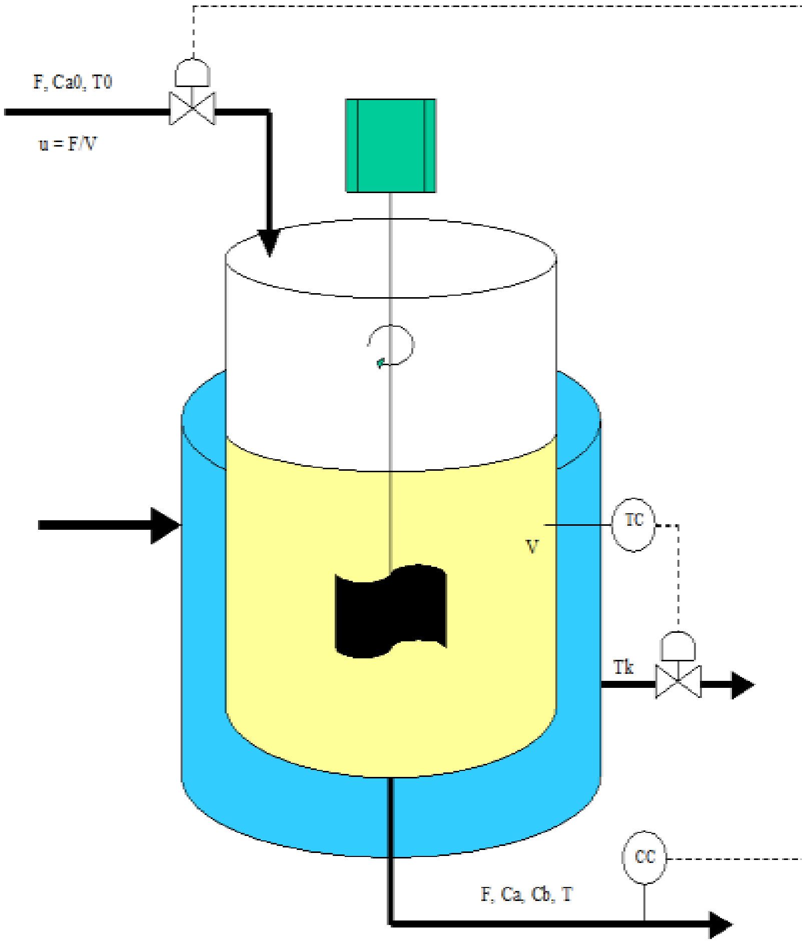
(Klatt and Engell, 1998)

- Van de Vusse reaction:

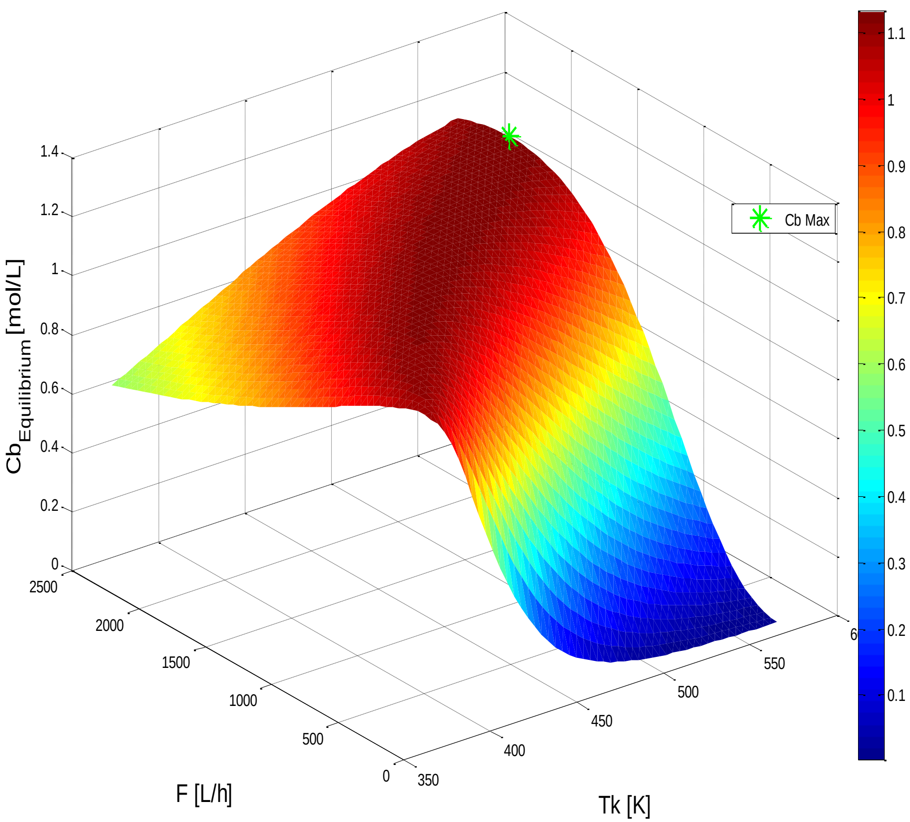
$$A \xrightarrow{k_1} B \xrightarrow{k_2} C \quad 2A \xrightarrow{k_3} D$$

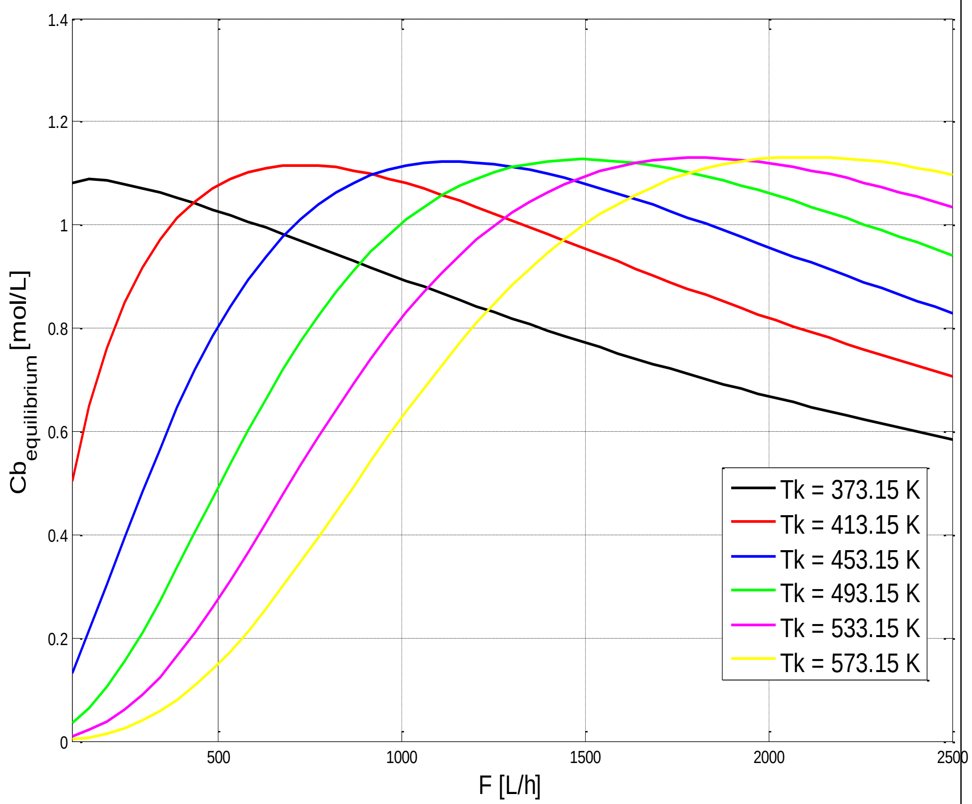
- Control objectives (in priority order)
  1) Bring F_B to its set point F_B^{SP}
  2) Maximize C_B

- Manipulated variables:
  - Inlet flow F
  - Coolant temperature in the reactor jacket T_k



Steady State Open loop behaviour of the VDV reactor



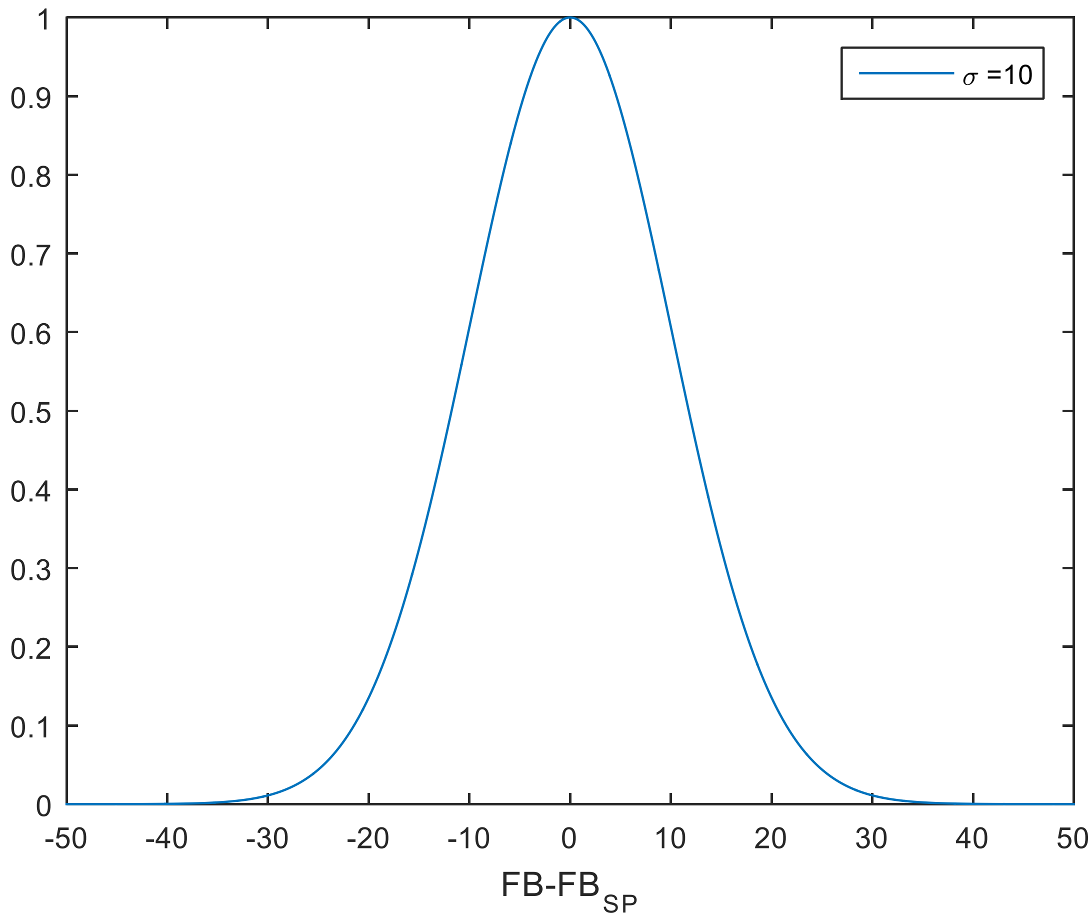


Immediate reward

$$r_t = C_B(t+1)\, e^{-\frac{1}{2\sigma^2}(F_B(t+1)-F_B^{SP})^2} - C_B(t)\, e^{-\frac{1}{2\sigma^2}(F_B(t)-F_B^{SP})^2}$$

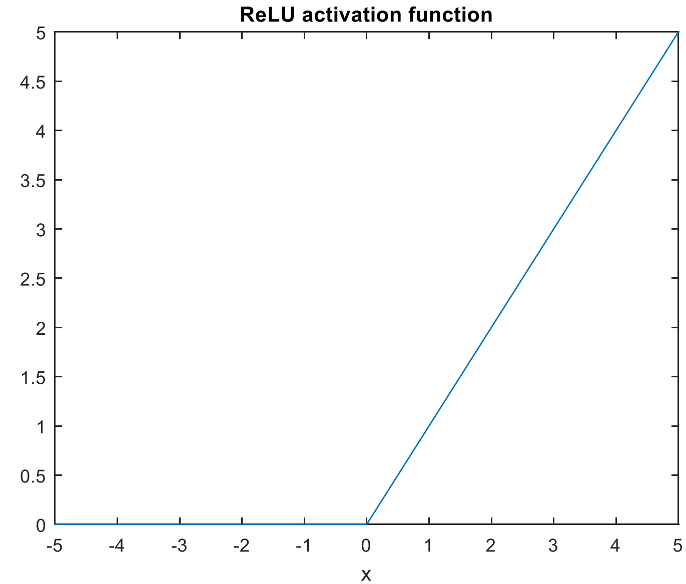
Gaussian membership function

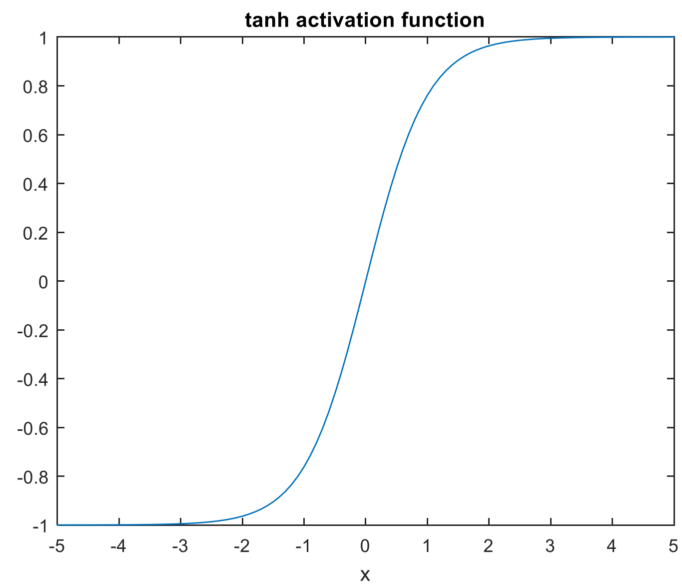
$$e^{-\frac{1}{2\sigma^2}(F_B(t)-F_B^{SP})^2}$$



Actor and critic neural networks (Lillicrap et al., 2016)

- Actor network:
  - Fully connected neural networks
  - 2 hidden layers of 400 and 300 neurons with Rectified Linear Unit (ReLU) activation function
  - Output layer with hyperbolic tangent (tanh) activation function to bound the input efforts
- Critic network:
  - Fully connected neural networks
  - 2 hidden layers of 400 and 300 neurons with Rectified Linear Unit (ReLU) activation function
  - Linear Output layer





Training

- Training through simulated experiments with random initial steady state
  - Sampling time: 0.01 hours
  - Maximum time for an experiment: 100 hours
  - Total time for training: 15000 hours
- Feed initial conditions

Variable	Value
Tin (°C)	110
CA,in (mol/L)	5,1
CB,in	0

Action bounds

Variable	Value
$\|\Delta F\|$	1 L/h
$\|\Delta Tk\|$	0.1 °C

- Set point for outlet B molar flow: 2000 mol/h
- Ornstein-Uhlenbeck process model used for exploration noise (Lillicrap et al., 2016)

Simulation results

- Initial steady state:

Inlet variables	Value
Tin (ºC)	110
CA,in (mol/L)	5,1
CB,in	0

Manipulated variables	Value
F (L/h)	1000
Tk (°C)	100
CB,in	0

Controlled variables	Value
FB (L/h)	457,03
CB (mol/L)	0,457

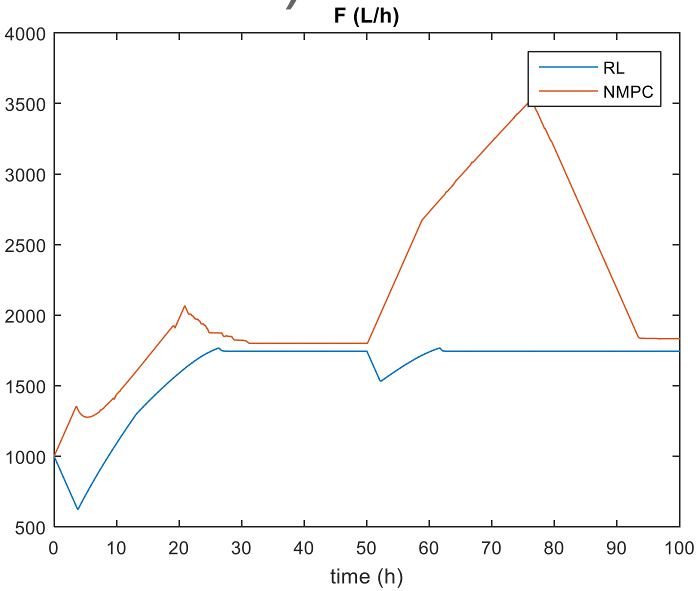
- T_{in} is considered as an unmeasured disturbance

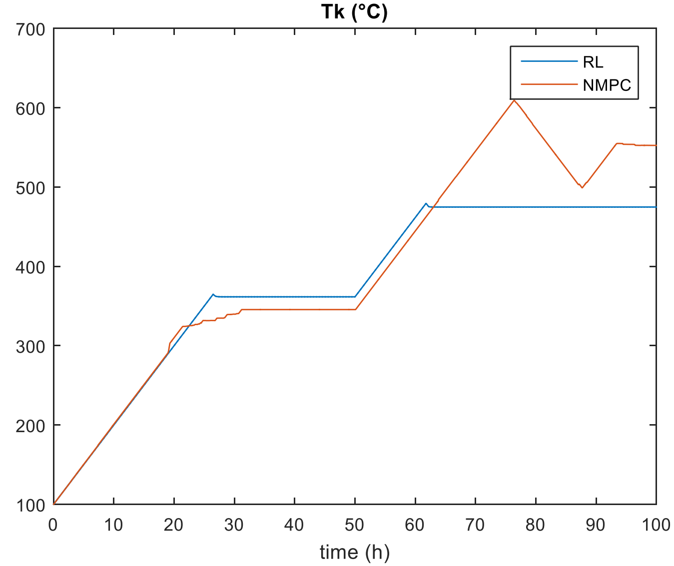
- Compararison with classical NMPC
  - Objective function $J(t)=\sum_{k=1}^{p} 1000\left(\frac{F_B(t+k)-F_B^{SP}}{1000}\right)^2+\left(\frac{C_B(t+k)-C_B^{SP}}{1.5}\right)^2$

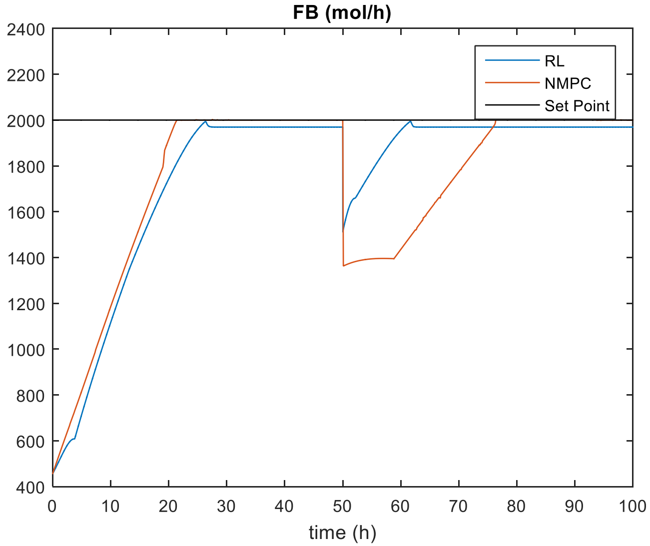
with $C_B^{SP}=1.2\,\mathrm{mol/L}$

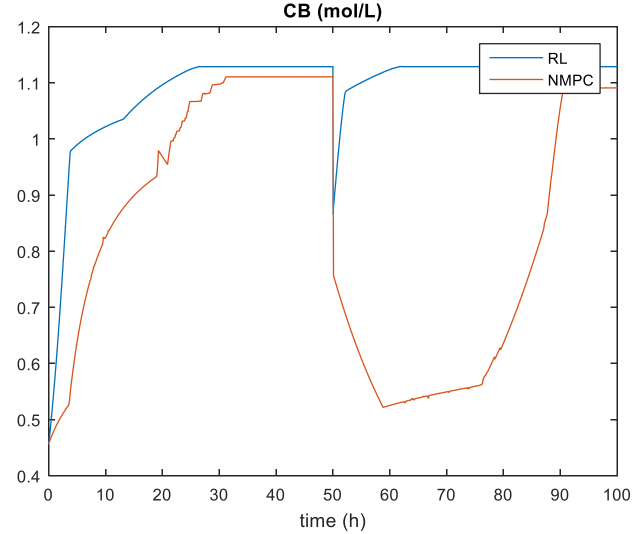
  - Sampling time: 0.1 hora
  - Control horizon m=5, Prediction horizon p=100

Simulations results (T_{in}: 110→90°C at t=50)

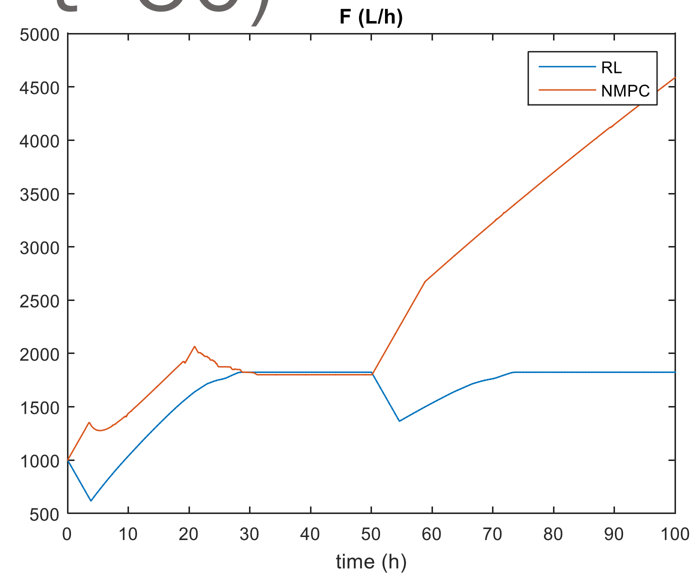


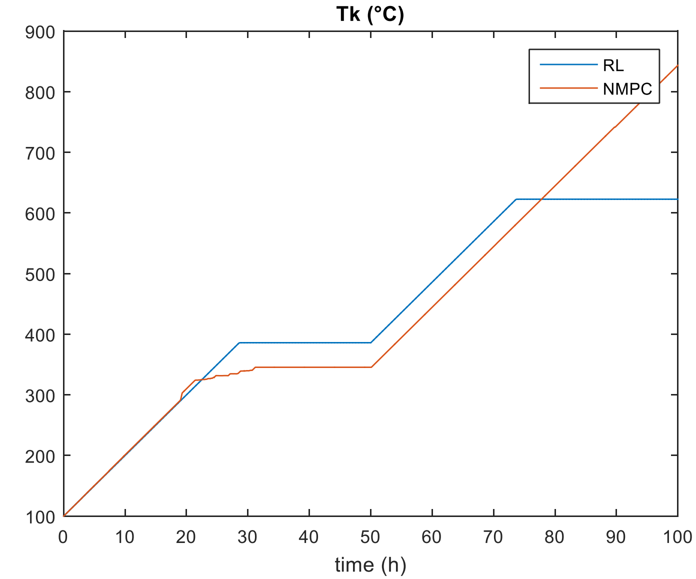


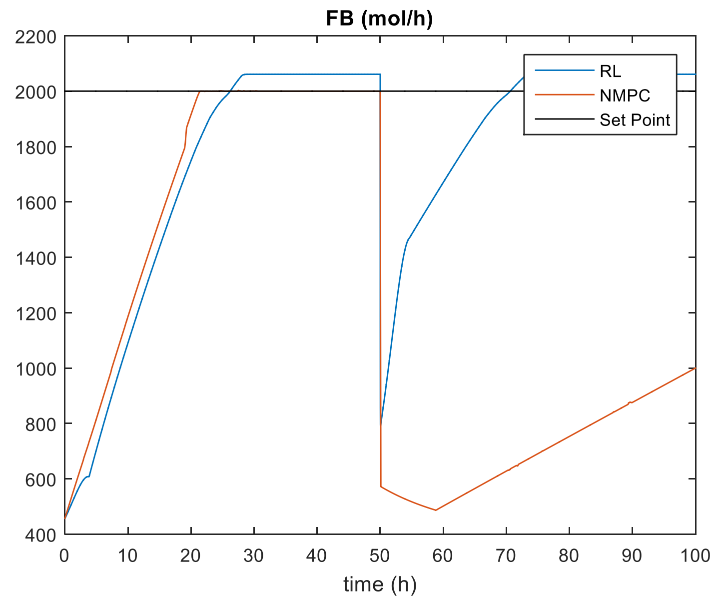


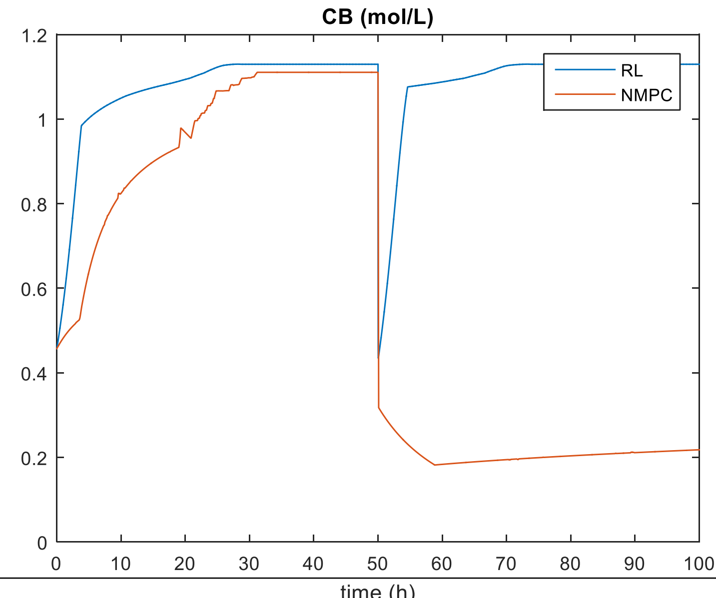


Simulations results (T_{in}: 110→70°C at t=50)









Conclusions

- A deep Reinforcement Learning based controller was implemented for the control and optimization of the VDV reactor
- The proposed controller showed a good performance compared to a classical NMPC controller.
- RL-based techniques are promising tools for the control and optimization of chemical processes.

Thank you!

References

- Charles W Anderson, Douglas C Hittle, Alon D Katz, and R Matt Kretchmar. Synthesis of reinforcement learning, neural networks and pi control applied to a simulated heating coil. Articial Intelligence in Engineering, 11(4):421-429, 1997.
- Felix Hernandez-del Olmo, Elena Gaudioso, Raquel Dormido, and Natividad Duro. Tackling the start-up of a reinforcement learning agent for the control of wastewater treatment plants. Knowledge-Based Systems, 2017.
- JC Hoskins and DM Himmelblau. Process control via articial neural networks and reinforcement learning. Computers & chemical engineering, 16(4):241-251, 1992.
- KLATT and ENGELL, Gain-scheduling trajectory control of a continuous stirred tank reactor. Computers and Chemical Engineering, vol.22,No. 4/5, p. 491-502, 1998.
- TP Lillicrap, JJ Hunt, A Pritzel, N Heess, T Erez, Y Tassa, D Silver, and D Wierstra. Continous Control with deep reinforcement learning. ICRL conference paper, 2016.
- Y Ma, W Zhu, MG Benton and J Romagnoli, Continuous control of a polymerization system with deep reinforcement learning. Journal of Process Control, Vol. 75, p. 40-47, 2019.
- EC Martinez. Batch process modeling for optimization using reinforcement learning. Computers & Chemical Engineering, 24(2-7):1187-1193, 2000.

References

- Dazi Li, Li Qian, Qibing Jin, and Tianwei Tan. Reinforcement learning control with adaptive gain for a saccharomyces cerevisiae fermentation process. Applied Soft Computing, 11(8):4488-4495, 2011.
- MA Mustafa and JA Wilson. Application of reinforcement learning to batch distillation. 2012.
- BJ Pandian and MMNoel, Control of a bioreactor using a new partially supervised reinforcement learning algorithm. Journal of Process Control, Vol. 69, p. 16-29, 2018.
- Prabhu Ramanathan, Kamal Kant Mangla, and Sitanshu Satpathy. Smart controller for conical tank system using reinforcement learning algorithm. Measurement, 2017.
- Hitesh Shah and M Gopal. Model-free predictive control of nonlinear processes based on reinforcement learning. IFAC-PapersOnLine, 49(1):89-94, 2016.
- Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction, volume 1. MIT press Cambridge, 1998.
- S Syafiie, F Tadeo, and E Martinez. Model-free learning control of chemical processes. In Reinforcement Learning. InTech, 2008.
- XI Yu-Geng, LI De-Wei, and Lin Shu. Model predictive control status and challenges. Acta Automatica Sinica, 39(3):222-236, 2013