

Model Predictive Control and Reinforcement Learning

– Fall School 2026: Trial Exam –

Each question has exactly one correct answer. Unless stated otherwise, we use the notation of the school: states s , actions a , stage cost $\ell(s, a)$ that is *minimized*, discount factor γ and transition model P . Questions on Lectures 05 and 06 use rewards R that are *maximized*, as in those lectures.

Lecture 01: Dynamic Systems and Simulation

1. One step of the explicit Euler integrator with step size h for the ODE $\dot{s} = f_c(s, a)$ is
 - (A) $s^+ = s + h f_c(s, a)$
 - (B) $s^+ = s + f_c(s, a)$
 - (C) $s^+ = h f_c(s, a)$
 - (D) $s^+ = s - h f_c(s, a)$
2. Consider $\dot{s} = -2s$ with $s(0) = 1$. What does one explicit Euler step with $h = 0.25$ give?
 - (A) 1.5
 - (B) $e^{-0.5}$
 - (C) 0.75
 - (D) 0.5
3. What is the order of the classical Runge–Kutta method RK4?
 - (A) 1
 - (B) 4
 - (C) 2
 - (D) 8
4. You halve the step size h of an RK4 integrator. Approximately by which factor does the global integration error decrease?
 - (A) 2
 - (B) 4
 - (C) 16
 - (D) It does not change
5. Which integrators are usually better suited for *stiff* systems?
 - (A) Implicit integrators
 - (B) Explicit integrators
 - (C) The explicit Euler method with a large step size
 - (D) No integrator can simulate stiff systems
6. How are collocation methods related to Runge–Kutta methods?
 - (A) They are a special class of explicit Runge–Kutta methods
 - (B) They are unrelated to Runge–Kutta methods
 - (C) They can only be used for linear ODEs
 - (D) They are a special class of implicit Runge–Kutta methods

7. When we transform a continuous-time system to discrete time with a *zero-order hold*, the control $a(t)$ is
- (A) zero on every interval
 - (B) constant on every interval $[t_k, t_{k+1}]$
 - (C) linear in time on every interval
 - (D) chosen by the integrator
8. Which is a suitable state vector for a simple pendulum with angle θ ?
- (A) $s = \theta$
 - (B) $s = \ddot{\theta}$
 - (C) $s = (\theta, \dot{\theta})$
 - (D) $s = (\dot{\theta}, \ddot{\theta})$
9. In practice, the sensors only measure a part of the state. What does an MPC controller then additionally need?
- (A) A larger prediction horizon
 - (B) A discount factor $\gamma < 1$
 - (C) A neural network
 - (D) A state estimator

Lecture 02: Numerical Optimization

10. Which identity is correct?
- (A) $\max_w F(w) = \min_w F(w)$
 - (B) $\max_w F(w) = -\min_w (-F(w))$
 - (C) $\max_w F(w) = \min_w (-F(w))$
 - (D) $\max_w F(w) = 1/\min_w F(w)$
11. Let F be differentiable and w^* a local minimizer of the *unconstrained* problem $\min_w F(w)$. What must hold?
- (A) $F(w^*) = 0$
 - (B) $\nabla^2 F(w^*) = 0$
 - (C) $\nabla F(w^*) = 0$
 - (D) $\nabla F(w^*) > 0$
12. Assume $\nabla F(w^*) = 0$ and $\nabla^2 F(w^*) \succ 0$ (positive definite). Then w^* is
- (A) a strict local minimizer
 - (B) a strict local maximizer
 - (C) a saddle point
 - (D) infeasible
13. The point $w = (0, 0)$ of $F(w) = w_1^2 - w_2^2$ is
- (A) a local minimizer
 - (B) a local maximizer
 - (C) a global minimizer
 - (D) a saddle point

14. Which function $F : \mathbb{R} \rightarrow \mathbb{R}$ is convex?

- (A) $F(w) = -w^2$
- (B) $F(w) = w^2$
- (C) $F(w) = \sin(w)$
- (D) $F(w) = w^3$

15. What is special about a *convex* optimization problem?

- (A) Every local minimizer is a global minimizer
- (B) It always has a unique solution
- (C) It can only have equality constraints
- (D) It can only be solved with Newton's method

16. The problem $\min_w g^\top w$ s.t. $Aw - b = 0, Cw - d \geq 0$ is a

- (A) Quadratic Program (QP)
- (B) Mixed-Integer Program (MIP)
- (C) Linear Program (LP)
- (D) Nonlinear Program (NLP) but not an LP

17. Linear MPC with linear dynamics, quadratic costs and linear constraints leads to which problem class?

- (A) Linear Program (LP)
- (B) Mixed-Integer Program (MIP)
- (C) Semidefinite Program (SDP)
- (D) Quadratic Program (QP)

18. The Newton step p_k for $\min_w F(w)$ at the iterate w_k is

- (A) $p_k = -\nabla F(w_k)$
- (B) $p_k = -(\nabla^2 F(w_k))^{-1} \nabla F(w_k)$
- (C) $p_k = -\nabla^2 F(w_k) \nabla F(w_k)$
- (D) $p_k = -F(w_k) / \nabla F(w_k)$

19. What is the local convergence rate of Newton's method (close to a solution with positive definite Hessian)?

- (A) quadratic
- (B) linear
- (C) sublinear
- (D) It does not converge

Lecture 03: MDPs, Dynamic Programming and LQR

20. An MDP is the tuple $(\mathcal{S}, \mathcal{A}, P, \ell, \gamma)$. Which of the following is *not* part of an MDP?

- (A) The set of actions $\mathcal{A}$
- (B) The transition model P
- (C) The policy π
- (D) The discount factor γ

- 21.** The Markov property says that
- (A) the next state depends only on the current state and action
 - (B) the next state depends on the whole history of states
 - (C) the system is deterministic
 - (D) the cost is always positive
- 22.** An episode ends after three steps with stage costs 2, 2 and 4. What is the return G_0 for $\gamma = 0.5$?
- (A) 8
 - (B) 5.5
 - (C) 3
 - (D) 4
- 23.** Why do we often use a discount factor $\gamma < 1$ in infinite-horizon problems?
- (A) To make the dynamics linear
 - (B) To keep the cumulative cost finite
 - (C) To make the problem deterministic
 - (D) To make the policy stochastic
- 24.** Standard RL maximizes a reward R_k . How does it relate to the stage cost ℓ used in this school?
- (A) $R_k = \ell(S_k, A_k)^2$
 - (B) $R_k = 1/\ell(S_k, A_k)$
 - (C) $R_k = -\ell(S_k, A_k)$
 - (D) $R_k = \gamma \ell(S_k, A_k)$
- 25.** Bellman's principle of optimality states:
- (A) The remaining part of an optimal plan must itself be optimal from the state it reaches
 - (B) The first action of an optimal plan is always zero
 - (C) An optimal plan never visits the same state twice
 - (D) A greedy policy is always optimal
- 26.** How is finite-horizon dynamic programming computed?
- (A) Forward in time, starting from $V_0 = 0$
 - (B) By random sampling of trajectories
 - (C) By gradient descent on the policy
 - (D) Backward in time, starting from the terminal cost $V_N = \bar{V}$
- 27.** You represent a value function on a grid with $m = 10$ points in each of $n_s = 6$ state dimensions. How many grid points do you need?
- (A) 60
 - (B) 10^6
 - (C) 6^{10}
 - (D) 16

- 28.** Consider a deterministic problem with $\gamma = 1$. In state s , action a_1 costs 1 and leads to a state with value 4. Action a_2 costs 3 and leads to a state with value 1. What does one dynamic programming (Bellman) update give?
- (A) $V(s) = 5$, greedy action a_1
 - (B) $V(s) = 1$, greedy action a_1
 - (C) $V(s) = 4$, greedy action a_2
 - (D) $V(s) = 9$, greedy action a_2
- 29.** For LQR (linear dynamics, quadratic cost), the optimal value function and policy have the form
- (A) $V^*(s) = s^\top X s$ and $\mu^*(s) = -K s$
 - (B) $V^*(s) = c^\top s$ and $\mu^*(s) = -K s$
 - (C) $V^*(s) = s^\top X s$ and $\mu^*(s) = \text{sign}(s)$
 - (D) V^* and μ^* have no closed form and need a neural network
- 30.** What does the Riccati recursion propagate backward in time?
- (A) The state trajectory s_k
 - (B) The value function evaluated on a grid
 - (C) The discount factor γ
 - (D) The matrix X_k of the quadratic value function
- 31.** Policy iteration alternates between
- (A) exploration and exploitation
 - (B) policy evaluation and policy improvement
 - (C) forward and backward simulation
 - (D) collecting data and discarding data
- 32.** Value iteration repeatedly applies
- (A) the Bellman *optimality* update $V \leftarrow \min_a \{ \ell(s, a) + \gamma \sum_{s'} P(s'|s, a) V(s') \}$
 - (B) a gradient step on the policy parameters
 - (C) the Riccati recursion
 - (D) a Monte Carlo average of returns

Lecture 04: Monte Carlo and Temporal-Difference Learning

- 33.** What do Monte Carlo (MC) methods for RL need?
- (A) The transition model P
 - (B) The optimal value function V^*
 - (C) A linear system
 - (D) Sampled episodes of experience
- 34.** MC prediction estimates $V^\pi(s)$ by
- (A) solving the Bellman equation with the known model
 - (B) averaging the returns observed after visiting s
 - (C) taking the minimum of all observed costs
 - (D) using the TD target

- 35.** In TD learning, *bootstrapping* means
- (A) updating an estimate based on other estimates
 - (B) restarting the episode from the initial state
 - (C) using a replay buffer
 - (D) sampling with replacement from the data
- 36.** The TD(0) target for $V(S_k)$ is
- (A) G_k
 - (B) $V(S_{k+1})$
 - (C) $\ell(S_k, A_k) + \gamma V(S_{k+1})$
 - (D) $\ell(S_k, A_k) - V(S_k)$
- 37.** Let $V(S) = 2$, $V(S') = 3$, $\ell(S, A) = 1$, $\gamma = 1$ and step size $\alpha = 0.5$. What is $V(S)$ after one TD(0) update?
- (A) 2.5
 - (B) 3
 - (C) 4
 - (D) 1.5
- 38.** Compare the MC target with the TD target. Which statement is typically true?
- (A) MC targets are biased and have low variance
 - (B) TD targets are unbiased and have high variance
 - (C) Both have zero variance
 - (D) MC targets are unbiased but have higher variance; TD targets have lower variance but can be biased
- 39.** An ϵ -greedy policy with $\epsilon = 0.2$ chooses among 4 actions. With which probability does it take the greedy action?
- (A) 0.8
 - (B) 0.2
 - (C) 0.85
 - (D) 0.95
- 40.** Which is the Q-learning update (with costs)?
- (A) $Q(S, A) \leftarrow Q(S, A) + \alpha [\ell(S, A) + \gamma \min_{a'} Q(S', a') - Q(S, A)]$
 - (B) $Q(S, A) \leftarrow Q(S, A) + \alpha [\ell(S, A) + \gamma Q(S, A) - Q(S', A')]$
 - (C) $Q(S, A) \leftarrow \ell(S, A)$
 - (D) $Q(S, A) \leftarrow Q(S, A) + \alpha [\ell(S, A) - Q(S, A)]^2$
- 41.** Let $Q(S, A) = 0$, $\ell(S, A) = 2$, $\gamma = 0.5$, $\alpha = 0.5$, and in the next state $Q(S', a_1) = 2$ and $Q(S', a_2) = 4$. What is $Q(S, A)$ after one Q-learning update (with costs)?
- (A) 3
 - (B) 2
 - (C) 1.5
 - (D) 2.5

42. Why is Q-learning an off-policy method?

- (A) Because it uses a neural network
- (B) Because it learns the values of the greedy policy while acting with another (e.g. ϵ -greedy) policy
- (C) Because it waits until the end of the episode
- (D) Because it does not use a step size

43. Off-policy MC weights returns with the importance sampling ratio $\rho = \prod_i \pi(A_i|S_i)/b(A_i|S_i)$. What is *not* needed to compute ρ ?

- (A) The target policy π
- (B) The behaviour policy b
- (C) The actions that were taken
- (D) The transition model P

Lecture 05: RL with Function Approximation

44. Why do we use function approximation in RL?

- (A) Because tables are infeasible for very large or continuous state spaces
- (B) Because it makes the MDP deterministic
- (C) Because it removes the need for exploration
- (D) Because it always guarantees convergence

45. For a linear value function $\hat{v}(s, \mathbf{w}) = \mathbf{w}^\top \mathbf{x}(s)$ with features $\mathbf{x}(s)$, the gradient $\nabla_{\mathbf{w}} \hat{v}(s, \mathbf{w})$ is

- (A) $\mathbf{w}$
- (B) $\mathbf{x}(s)$
- (C) $\mathbf{w}^\top \mathbf{x}(s)$
- (D) 0

46. Why are TD methods with function approximation called *semi-gradient* methods?

- (A) They only update half of the weights
- (B) They use half the step size
- (C) The gradient is taken only through the prediction, not through the bootstrapped target
- (D) They only work for linear features

47. The *deadly triad* that can cause instability and divergence is

- (A) exploration, exploitation and discounting
- (B) on-policy learning, Monte Carlo and tables
- (C) model learning, planning and acting
- (D) function approximation, bootstrapping and off-policy learning

48. Why does DQN use an experience replay buffer?

- (A) To reduce the correlation between consecutive samples and to reuse data
- (B) To store the optimal policy
- (C) To make the environment deterministic
- (D) To avoid the need for a discount factor

49. Why does DQN use a target network with frozen parameters $\mathbf{w}^-$?

- (A) To increase exploration
- (B) To compute the actions faster
- (C) To store a copy of the environment
- (D) To keep the targets stable and avoid oscillations during training

50. A “soft” target network update with $\tau \in (0, 1]$ is

- (A) $\mathbf{w}^- \leftarrow \mathbf{w}^- + \tau$
- (B) $\mathbf{w}^- \leftarrow (1 - \tau) \mathbf{w}^- + \tau \mathbf{w}$
- (C) $\mathbf{w} \leftarrow \mathbf{w}^-$ in every step
- (D) $\mathbf{w}^- \leftarrow \tau \mathbf{w}^-$

51. What does the Q-network in DQN output for Atari games?

- (A) The next image
- (B) The probability of winning
- (C) One Q-value for each of the discrete actions
- (D) A continuous action

Lecture 06: Actor-Critic Methods

52. Policy gradient methods

- (A) update the parameters θ of a policy π_θ by gradient ascent on the performance $J(\theta)$
- (B) only learn a value function
- (C) require the transition model P
- (D) only work for discrete states

53. What is $\nabla_\theta \log \pi_\theta(a|s)$ called?

- (A) The TD error
- (B) The advantage
- (C) The score function
- (D) The entropy

54. REINFORCE uses Monte Carlo returns. Its gradient estimates are therefore

- (A) biased and low variance
- (B) unbiased, but can have high variance
- (C) always zero
- (D) deterministic

55. A baseline b in policy gradient methods keeps the gradient estimate unbiased if it depends

- (A) on the action only
- (B) on the state and the action
- (C) on the next action
- (D) only on the state (not on the action)

- 56.** In actor-critic methods,
- (A) the actor is the learned policy and the critic is a learned value function
 - (B) the actor is a learned value function and the critic is the policy
 - (C) the actor is the environment and the critic is the reward
 - (D) the critic chooses the actions
- 57.** Compared to REINFORCE with baseline, a one-step actor-critic replaces the full return G_t by
- (A) zero
 - (B) the immediate reward R_{t+1} only
 - (C) the bootstrapped target $R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w})$
 - (D) the maximum reward in the episode
- 58.** What is the purpose of the clipped surrogate objective in PPO?
- (A) To keep the new policy close to the old policy
 - (B) To clip the rewards to $[-1, 1]$
 - (C) To make the policy deterministic
 - (D) To remove the need for a critic
- 59.** How does DDPG handle the maximization over continuous actions $\max_a Q(s, a)$?
- (A) It discretizes the action space
 - (B) It solves an MPC problem in every step
 - (C) It samples random actions and takes the best
 - (D) A deterministic actor $\mu_\theta(s)$ is trained to approximate $\arg \max_a Q(s, a)$
- 60.** How does DDPG explore?
- (A) By using an ϵ -greedy policy over all actions
 - (B) By adding (Gaussian) noise to the output of the actor
 - (C) By maximizing the entropy
 - (D) It does not explore
- 61.** What does Soft Actor-Critic (SAC) maximize?
- (A) Only the expected return
 - (B) Only the policy entropy
 - (C) The expected return plus an entropy bonus weighted by α
 - (D) The TD error

Lecture 07: Nonlinear Optimization for MPC

- 62.** Direct optimal control methods (e.g. multiple shooting, collocation) follow the principle
- (A) first discretize, then optimize
 - (B) first optimize, then discretize
 - (C) first learn, then optimize
 - (D) only simulate, never optimize

- 63.** In direct *multiple shooting*, the states $x_0, \dots, x_N$ and controls $u_0, \dots, u_{N-1}$ are optimization variables. How many variables are there with n_x states and n_u controls?
- (A) Nn_u
 - (B) Nn_x
 - (C) $(N+1)n_x + Nn_u$
 - (D) $n_x + n_u$
- 64.** In direct *single shooting*, the states are eliminated by simulation. How many optimization variables remain (piecewise-constant controls, fixed initial state)?
- (A) Nn_u
 - (B) $(N+1)n_x + Nn_u$
 - (C) Nn_x
 - (D) N^2n_u
- 65.** A constraint $H_i(w) \geq 0$ is inactive at the solution, $H_i(w^*) > 0$. By complementary slackness, its multiplier is
- (A) $\mu_i^* > 0$
 - (B) $\mu_i^* < 0$
 - (C) $\mu_i^* = 1$
 - (D) $\mu_i^* = 0$
- 66.** A feasible point that satisfies the KKT conditions of a general (nonconvex) NLP is
- (A) always the global minimizer
 - (B) a candidate for a local minimizer
 - (C) always a boundary point
 - (D) never a minimizer
- 67.** Sequential Quadratic Programming (SQP) solves an NLP by
- (A) solving a sequence of quadratic programs
 - (B) enumerating all active sets at once
 - (C) random search
 - (D) solving one linear program
- 68.** To which problems can the Gauss–Newton Hessian approximation be applied?
- (A) Only linear programs
 - (B) Any nonconvex objective
 - (C) Only problems without constraints
 - (D) Problems with a (nonlinear) least-squares objective
- 69.** Interior-point methods replace the constraints $H_i(w) \geq 0$ by the barrier term $-\tau \sum_i \log H_i(w)$ in the objective. What happens to τ during the algorithm?
- (A) τ is increased to infinity
 - (B) τ is kept at 1
 - (C) τ is driven towards 0
 - (D) τ is chosen randomly

- 70.** Consider an MPC problem with horizon $N = 10$ for the system $s^+ = 2s^2 + a$, $a \in [-1, 1]$, stage cost $\ell(s, a) = a^2$ and terminal cost $100s^2$. To which class does it belong?
- (A) Linear Program (LP)
 - (B) Mixed-Integer Program (MIP)
 - (C) Quadratic Program (QP) but not LP
 - (D) Nonlinear Program (NLP) but not QP

Lecture 08: Differentiable MPC

- 71.** In differentiable MPC, the *solution sensitivity* is
- (A) the derivative $\partial z^*/\partial \phi$ of the optimal solution with respect to the parameters ϕ
 - (B) the derivative of the dynamics with respect to time
 - (C) how sensitive the solver is to rounding errors
 - (D) the number of iterations needed
- 72.** Which theorem is the basis for computing solution sensitivities from the KKT conditions?
- (A) The policy gradient theorem
 - (B) The implicit function theorem
 - (C) The central limit theorem
 - (D) Bellman's principle of optimality
- 73.** Consider $\min_x (x - \phi^2)^2$ s.t. $-1 \leq x \leq 1$. What are $x^*(\phi)$ and $dx^*/d\phi$ at $\phi = 2$?
- (A) $x^* = 4$, $dx^*/d\phi = 4$
 - (B) $x^* = 1$, $dx^*/d\phi = 4$
 - (C) $x^* = 1$, $dx^*/d\phi = 0$
 - (D) $x^* = -1$, $dx^*/d\phi = 0$
- 74.** Why are the interior-point KKT conditions with a barrier parameter $\tau > 0$ (smoothing) useful for differentiation?
- (A) They make the problem convex
 - (B) They remove all constraints
 - (C) They make the solver deterministic
 - (D) The exact solution map has kinks at active-set changes; smoothing makes it differentiable
- 75.** You have one scalar loss and many parameters, as in backpropagation. Which sensitivities are most efficient?
- (A) Adjoint sensitivities (vector-Jacobian products)
 - (B) Forward sensitivities of the full Jacobian
 - (C) Finite differences in each parameter
 - (D) No sensitivities are needed
- 76.** acados uses a nominal solver and a separate sensitivity solver. Why does the sensitivity solver evaluate the *exact* Hessian?
- (A) Because the exact Hessian is cheaper than Gauss-Newton
 - (B) Because sensitivities computed with an inexact Hessian (e.g. Gauss-Newton) are generally wrong
 - (C) Because the nominal solver cannot handle constraints
 - (D) Because PyTorch requires it

acados

77. In which language is the core of **acados** written?

- (A) Julia
- (B) Python
- (C) C
- (D) MATLAB

78. Which of the following is a QP solver that **acados** uses (e.g. for its Riccati-based linear algebra)?

- (A) IPOPT
- (B) PyTorch
- (C) CasADi
- (D) HPIPM

79. Which NLP solution methods does **acados** offer for optimal control problems?

- (A) SQP and Real-Time Iterations (RTI)
- (B) Genetic algorithms
- (C) Branch and bound
- (D) The simplex method

80. What does a Real-Time Iteration (RTI) do at each sampling time?

- (A) It solves the NLP to full convergence
- (B) It performs one SQP iteration, i.e. it solves one QP
- (C) It only simulates the system
- (D) It retrains a neural network

81. What is the main reason why **acados** is fast?

- (A) It runs only on GPUs
- (B) It ignores the constraints
- (C) It exploits the stage-wise (block-sparse) structure of optimal control problems
- (D) It uses finite differences for all derivatives

82. Which integrators does **acados** provide?

- (A) Only the explicit Euler method
- (B) No integrators; the user must provide the discrete dynamics
- (C) Only symbolic solutions of the ODE
- (D) Explicit and implicit Runge–Kutta methods for ODEs and DAEs

83. In the cart-pole exercise, you wanted to reduce the computation time of the **acados** OCP solver. Which change does *not* help?

- (A) Using real-time iterations
- (B) Increasing the horizon length N a lot
- (C) Looser QP solver tolerances
- (D) Partial condensing

84. How was model-plant mismatch introduced in the cart-pole exercise?
- (A) The cart mass in the simulator was increased by 10%, but not in the MPC model
 - (B) The MPC horizon was set to $N = 1$
 - (C) The cost weights were set to zero
 - (D) The actuator limits were removed

Lecture 09: Synthesis of MPC and RL

85. Which comparison between MPC and RL is typically correct?
- (A) MPC has low online computation time, RL has high online computation time
 - (B) MPC needs no model, RL needs an exact model
 - (C) MPC has high online computation time, RL only needs a forward pass of the policy
 - (D) Both need the same online computation time
86. How are constraints typically handled?
- (A) In RL constraints are inherent, in MPC they are hard to include
 - (B) Neither can handle constraints
 - (C) Both only handle constraints through penalties
 - (D) In MPC constraints are handled directly, in RL they are hard to handle formally
87. The system dynamics change. What do you typically have to do?
- (A) MPC: update the model f ; RL: collect new data and retrain the policy
 - (B) MPC: retrain a neural network; RL: nothing
 - (C) Both: nothing
 - (D) MPC: change the discount factor; RL: change the step size
88. In the *parameterized* architecture, the MPC parameters θ
- (A) are the output of a neural network evaluated at every time step
 - (B) are constant during deployment (learned offline), so no neural network is evaluated online
 - (C) change inside the optimization problem
 - (D) are chosen randomly
89. In the *hierarchical* architecture, a neural network
- (A) is embedded in the dynamics of the MPC
 - (B) replaces the MPC
 - (C) computes the MPC parameters $\phi = \varphi_\theta(s)$ before the MPC problem is solved
 - (D) is only used to plot results
90. Which is an example of the *integrated* architecture?
- (A) A learned neural network residual inside the MPC dynamics
 - (B) A neural network that outputs a reference trajectory before the MPC is solved
 - (C) A fixed, hand-tuned MPC
 - (D) A neural network that only warm-starts the solver

- 91.** In the *algorithmic* architecture, the learned component
- (A) changes the cost function
 - (B) replaces the dynamics model
 - (C) removes all constraints
 - (D) changes solver settings, e.g. provides an initial guess (warm start)
- 92.** When MPC is used as an *expert actor*,
- (A) the MPC is trained by RL
 - (B) a policy (e.g. a neural network) learns to imitate the actions of the MPC
 - (C) the MPC is only used as a safety filter
 - (D) the MPC replaces the environment
- 93.** How can an MPC be used to provide a Q-function $Q^{\text{MPC}}(s, a)$ (e.g. as a critic)?
- (A) By removing the dynamics constraints
 - (B) By setting the horizon to $N = 0$
 - (C) By additionally fixing the first control $u_0 = a$ in the MPC problem
 - (D) By maximizing instead of minimizing
- 94.** What does an MPC-based *safety filter* (postprocessing) do?
- (A) It projects the action of the RL policy to a safe action with respect to the model and the constraints
 - (B) It removes the constraints from the RL problem
 - (C) It generates training data for the RL policy
 - (D) It learns the value function
- 95.** What is the goal of *closed-loop learning*, as opposed to *MDP-aligned learning*?
- (A) To make the MPC model match the real system as accurately as possible
 - (B) To learn only the terminal cost by supervised learning
 - (C) To avoid using any data
 - (D) To tune the MPC parameters directly for the best closed-loop performance, even if the model becomes less accurate

Lectures 10–13: RL and MPC (Sébastien Gros)

- 96.** In the view of these lectures, a parameterized MPC can be seen as
- (A) a model of the solution of the MDP, i.e. we want $Q_\phi^{\text{MPC}} \approx Q^\star$ (or at least $\pi_\phi^{\text{MPC}} \approx \pi^\star$)
 - (B) a replacement for the MDP
 - (C) a method that only works for linear systems
 - (D) a neural network with fixed weights
- 97.** Is the model that best fits the data always the model that gives the best closed-loop performance of the MPC?
- (A) Yes, always
 - (B) No, not necessarily
 - (C) Yes, if the horizon is long enough
 - (D) Yes, if the system is deterministic

98. How is reinforcement learning used together with a parameterized MPC?

- (A) RL replaces the MPC by a deep neural network
- (B) RL is only used to identify the dynamics model
- (C) RL is a toolbox to adjust the MPC parameters ϕ for better closed-loop performance, e.g. via Q-learning or policy gradients
- (D) RL is not compatible with MPC

leap-c

99. What does `leap-c` provide?

- (A) A new QP solver written in Fortran
- (B) A simulator for Atari games
- (C) A replacement for `acados`
- (D) A differentiable MPC layer for PyTorch based on `acados`

100. You call the `leap-c` MPC layer with a given first action $u_0 = \bar{u}_0$ instead of `u0=None`. What value does it return?

- (A) The state-action value $Q^{\text{MPC}}(x_0, \bar{u}_0)$
- (B) The optimal first action u_0^*
- (C) The gradient of the loss
- (D) The policy entropy