Quiz and Review Questions on Model-Based Reinforcement Learning

Based on "A Comprehensive Review of MBRL" By Taha Majlesi

July 17, 2025

Contents

1 Multiple Choice Questions 2

2 Explanatory Questions 15

3 Answers to Multiple Choice Questions 17

4 Answers to Explanatory Questions 20

1 Multiple Choice Questions

1. What is the fundamental dichotomy in Reinforcement Learning discussed in the text?

   A. Policy-based vs. Value-based RL

   B. Model-free vs. Model-based RL

   C. On-policy vs. Off-policy RL

   D. Deterministic vs. Stochastic RL

2. In Model-Free RL (MFRL), how is the environment typically treated?

   A. As a known mathematical function

   B. As a "black box"

   C. As a deterministic system

   D. As a supervised learning problem

3. What is the primary goal of an agent in MFRL?

   A. To learn an explicit model of the environment's dynamics.

   B. To learn a policy or a value function directly.

   C. To minimize prediction error on state transitions.

   D. To perform "what-if" simulations internally.

4. Which component of the environment does MFRL explicitly make "no attempt to learn"?

   A. The reward function $r(s_t, a_t)$

   B. The policy $\pi_\theta(a_t|s_t)$

   C. The true transition dynamics $p(s_{t+1}|s_t, a_t)$

   D. The state space S

5. Model-Based RL (MBRL) separates the problem into which two distinct phases?

   A. Exploration and Exploitation

   B. Learning and Inference

   C. Model Learning and Planning

   D. Data Collection and Policy Evaluation

6. What is the model learning phase in MBRL analogous to?

   A. A human creating a "mental map"

   B. A direct trial-and-error process

   C. A simple reflex action

   D. A random search for rewards

7. How is the model learning phase in MBRL typically framed?

   A. As a reinforcement learning problem

   B. As a supervised learning problem

   C. As an unsupervised learning problem

   D. As a black-box optimization problem

8. What is the most significant advantage of MBRL over MFRL?

   A. Higher asymptotic performance

   B. Simpler computational profile

   C. Superior sample efficiency

   D. Faster decision-making at inference time

9. What phenomenon fundamentally caps the performance of an MBRL agent?

   A. The curse of dimensionality

   B. The exploration-exploitation trade-off

   C. Model bias

   D. Overfitting the policy

10. Why does MFRL often achieve higher asymptotic performance?

    A. It uses a more complex model.

    B. It learns directly from real experience and is not limited by model accuracy.

    C. It requires fewer samples to converge.

    D. It has a faster planning phase.

11. Which approach generally has a more complex and computationally demanding profile?

    A. Model-Free RL

    B. Model-Based RL

    C. Both are equally complex

    D. Neither is computationally demanding

12. How does MBRL typically exhibit greater adaptability to changes, such as a new reward function?

    A. By retraining the entire policy from scratch.

    B. By collecting a completely new dataset.

    C. By re-planning with the existing dynamics model.

    D. It is not more adaptable than MFRL.

13. What is ”model-induced” or ”uncertainty-driven” exploration?

    A. Exploring randomly to build a model.

B. Using a learned model to identify and explore uncertain regions of the state space.

C. Exploring only states with high rewards.

D. A strategy used exclusively in MFRL.

14. The Dyna-Q algorithm is an example of what?

A. A pure model-free algorithm

B. A pure model-based algorithm

C. A hybrid algorithm blending model-free and model-based elements

D. An algorithm that does not use a replay buffer

15. In a fully deterministic world, the planning problem becomes one of:

A. Monte Carlo simulation

B. Trajectory optimization

C. Q-learning

D. Policy gradient estimation

16. What is the fundamental flaw of open-loop planning in a stochastic world?

A. It is too computationally expensive.

B. It requires a perfect model.

C. It cannot react to unexpected outcomes.

D. It only works for discrete action spaces.

17. What does a closed-loop or feedback policy, $\pi(a_t|s_t)$, do?

A. It pre-computes a fixed sequence of actions.

B. It maps the current observed state to an action.

C. It only works in deterministic environments.

D. It ignores the current state.

18. Derivative-free planning methods are particularly useful when:

A. The model is a simple linear function.

B. Gradients of the model are unavailable or difficult to compute.

C. The action space is discrete.

D. The planning horizon is infinite.

19. What is the simplest ”guess and check” derivative-free planning method?

A. Cross-Entropy Method (CEM)

B. Monte Carlo Tree Search (MCTS)

C. Random Shooting

D. Policy Gradients

20. How does the Cross-Entropy Method (CEM) improve upon random shooting?

    A. By using model gradients.
    B. By iteratively refining a sampling distribution to focus on high-reward regions.
    C. By building a search tree.
    D. By using a model-free learner.

21. In CEM, what are the ”elite” samples used for?

    A. To be executed directly in the environment.
    B. To be discarded from the population.
    C. To refit the parameters of the sampling distribution.
    D. To calculate the model bias.

22. For a Gaussian sampling distribution in CEM, the refitting step involves calculating what from the elite set?

    A. The maximum likelihood estimate
    B. The sample mean and sample covariance
    C. The KL divergence
    D. The cross-entropy loss

23. Monte Carlo Tree Search (MCTS) is particularly well-suited for problems with:

    A. Continuous action spaces
    B. Unknown reward functions
    C. Discrete action spaces
    D. No model available

24. What is a ”rollout” or ”playout” in MCTS?

    A. The final action sequence chosen by the algorithm.
    B. A lightweight, randomized simulation from a node to the end of an episode.
    C. The process of expanding the tree.
    D. The backpropagation of value estimates.

25. Which of these is NOT one of the four main phases of an MCTS iteration?

    A. Selection
    B. Expansion
    C. Simulation
    D. Pruning

26. What is the purpose of the UCT formula in MCTS?

    A. To estimate the reward of a terminal state.

B. To balance exploration and exploitation during the selection phase.

C. To decide when to stop the search.

D. To update the model of the environment.

27. In the UCT formula, what does the term $Q(s_{t+1})/N(s_{t+1})$ represent?

A. The exploration bonus

B. The exploitation term (mean reward)

C. The visit count of the parent node

D. The probability of reaching the state

28. MCTS can be understood as a recursive application of which simpler RL problem?

A. The shortest path problem

B. The multi-armed bandit (MAB) problem

C. The linear regression problem

D. The knapsack problem

29. What is the primary reason a naive MBRL approach is "doomed to fail"?

A. Lack of data

B. Computational cost

C. Distributional shift

D. The use of neural networks

30. Distributional shift in MBRL refers to the mismatch between:

A. The model's predictions and reality.

B. The data distribution used for training the model and the distribution encountered by the planner.

C. The rewards in simulation and the rewards in the real world.

D. The policy's actions and the optimal actions.

31. What is the DAgger (Dataset Aggregation) algorithm adapted for RL designed to mitigate?

A. High computational cost

B. Distributional shift

C. The curse of dimensionality

D. Model bias in a static dataset

32. How does Model Predictive Control (MPC) combat the problem of compounding model errors?

A. By learning a more accurate model.

B. By planning over an infinite horizon.

C. By executing only the first action of a plan and then replanning from the new state.

D. By using a model-free algorithm as a backup.

33. What is the key idea behind modern, high-performance MBRL?

A. Using bigger neural networks.

B. Building models that explicitly quantify and reason about their own uncertainty.

C. Relying solely on model-free methods for planning.

D. Avoiding planning altogether.

34. What is aleatoric uncertainty?

A. Uncertainty due to the model's lack of knowledge.

B. Inherent, irreducible randomness in the environment.

C. Uncertainty that can be reduced by collecting more data.

D. The bias of the learned model.

35. What is epistemic uncertainty?

A. Uncertainty that arises from the model's own lack of knowledge due to limited data.

B. The inherent noise in sensor readings.

C. The stochasticity of the environment's dynamics.

D. Uncertainty that cannot be reduced with more data.

36. Which type of uncertainty can be reduced by collecting more relevant data?

A. Aleatoric

B. Epistemic

C. Both

D. Neither

37. High epistemic uncertainty should drive an agent towards what kind of behavior?

A. Caution and risk-aversion

B. Terminating the episode

C. Optimism and information-seeking exploration

D. Following a pre-computed plan

38. High aleatoric uncertainty signals what to the agent?

A. An opportunity to learn

B. A flaw in the model

C. Inherent, unavoidable risk

D. A region with no rewards

39. What is the core idea of the Bayesian modeling approach?

    A. To find a single best set of model parameters.
    B. To infer a full posterior distribution over the model parameters.
    C. To use ensembles of models.
    D. To eliminate uncertainty completely.

40. Bayesian Neural Networks (BNNs) learn what for each model parameter?

    A. A single point-estimate
    B. A probability distribution
    C. An upper confidence bound
    D. A Q-value

41. What is the most common approximate inference technique for training BNNs?

    A. Monte Carlo Tree Search
    B. Variational Inference (VI)
    C. Stochastic Gradient Descent
    D. Maximum Likelihood Estimation

42. What is a major drawback of BNNs?

    A. They cannot quantify uncertainty.
    B. They are computationally expensive and can be complex to train.
    C. They are less principled than deep ensembles.
    D. They can only be used for deterministic environments.

43. How does a deep ensemble quantify epistemic uncertainty?

    A. By the mean of the predictions across the ensemble.
    B. By the variance of the predictions across the ensemble.
    C. By using dropout at test time.
    D. By calculating the ELBO.

44. What is a primary advantage of deep ensembles over BNNs?

    A. They are more theoretically grounded.
    B. They require fewer model parameters.
    C. They are simpler to implement and highly parallelizable.
    D. They do not require random initialization.

45. In uncertainty-aware trajectory optimization, how is an action sequence evaluated?

    A. By a single rollout with the most accurate model.

B. By averaging the reward across rollouts from an ensemble of models.

C. By choosing the rollout with the maximum possible reward.

D. By ignoring the model and using a model-free evaluator.

46. The PETS algorithm combines which three key concepts?

A. MCTS, BNNs, and DAgger

B. Random Shooting, a single model, and open-loop planning

C. An ensemble model, MPC, and a sampling-based trajectory optimizer (CEM)

D. Q-learning, a linear model, and ϵ-greedy exploration

47. What does PETS stand for?

A. Policy Ensembles with Trajectory Search

B. Probabilistic Ensembles with Trajectory Sampling

C. Planning and Exploration with Thompson Sampling

D. Predictive Ensemble Trajectory Search

48. How can a planner be designed for uncertainty-driven exploration?

A. By penalizing epistemic uncertainty.

B. By augmenting the objective with an exploration bonus proportional to epistemic uncertainty.

C. By ignoring epistemic uncertainty entirely.

D. By only exploring states with known high rewards.

49. For safety-critical applications, how should a planner handle aleatoric uncertainty?

A. It should be ignored, as it cannot be reduced.

B. It should be treated as an exploration signal.

C. The objective should be modified to penalize it, leading to risk-averse behavior.

D. The objective should be modified to maximize it, to understand the worst-case scenarios.

50. The ”cheetah agent learning to fly” is an example of what failure mode?

A. The planner exploiting inaccuracies in a learned model on out-of-distribution states.

B. The agent being too risk-averse.

C. The agent failing to explore.

D. The computational cost of planning being too high.

51. What is the primary challenge for MFRL according to the comparison table?

A. Model bias

B. High sample complexity

C. Computationally intensive planning

D. Low adaptability

52. What is the primary challenge for MBRL according to the comparison table?

A. High sample complexity

B. Fast inference

C. Overcoming model bias and compounding errors

D. Low adaptability

53. A ”value-equivalent model” is trained to do what?

A. Predict future states with perfect accuracy.

B. Ensure plans rolled out in the model produce the same cumulative reward as in the real world.

C. Mimic the policy of a model-free agent.

D. Only predict the next immediate reward.

54. The Persian anecdote of the deaf man illustrates the failure of what?

A. Closed-loop planning

B. Model-based learning

C. Open-loop planning

D. Monte Carlo Tree Search

55. In the UCT formula, what is the role of the constant C?

A. It represents the cumulative reward.

B. It controls the degree of exploration.

C. It is the number of visits to a node.

D. It is a discount factor.

56. The backpropagation step in MCTS updates what?

A. The parameters of the learned dynamics model.

B. The policy network of the agent.

C. The visit counts and value estimates along the selection path.

D. The default rollout policy.

57. What does it mean for MCTS to be an ”anytime” algorithm?

A. It can be run at any time of day.

B. It can be run for any amount of time and will return the best action found so far.

C. It works for any type of environment.

D. It can be used with any model.

58. Which method is NOT a derivative-free optimization method?

    A. Random Shooting
    B. Cross-Entropy Method
    C. Trajectory optimization using calculus of variations
    D. MCTS

59. The ”refit distribution” step in CEM can be formally derived as minimizing what?

    A. The mean squared error between predicted and actual states.
    B. The cross-entropy between the elite distribution and the sampling distribution.
    C. The variance of the elite samples.
    D. The number of samples required for convergence.

60. What is a key benefit of MBRL's adaptability?

    A. It can change its neural network architecture on the fly.
    B. It can adapt to new tasks (e.g., new goal locations) by re-planning without new data.
    C. It adapts its learning rate automatically.
    D. It is guaranteed to find the optimal policy.

61. The performance of an MBRL agent is fundamentally ”capped” by what?

    A. The number of CPU cores available.
    B. The accuracy of its learned model.
    C. The size of its replay buffer.
    D. The complexity of the policy network.

62. Why is high sample efficiency critical in a domain like robotics?

    A. Robots have limited memory.
    B. Data collection can be expensive, time-consuming, or dangerous.
    C. Robotic simulators are always perfect.
    D. Robots can only perform discrete actions.

63. In the context of MBRL, what does ”planning” refer to?

    A. Collecting data from the environment.
    B. Training the dynamics model.
    C. Using the learned model to simulate outcomes and select actions.
    D. Updating the policy parameters with gradient descent.

64. What is the main difference between aleatoric and epistemic uncertainty?

    A. Aleatoric is reducible, epistemic is not.

B. Epistemic is reducible, aleatoric is not.

C. Aleatoric relates to rewards, epistemic relates to dynamics.

D. Epistemic relates to rewards, aleatoric relates to dynamics.

65. Monte Carlo Dropout, when used at test time, is an approximation of what?

A. A deep ensemble

B. A model-free Q-learning algorithm

C. Bayesian inference in a BNN

D. Model Predictive Control

66. What is the main drawback of deep ensembles?

A. They produce poor uncertainty estimates.

B. They are difficult to implement.

C. They have high computational cost due to training and running multiple models.

D. They cannot be used with MPC.

67. In the law of total variance for uncertainty decomposition, what does the epistemic term represent?

A. The expected noise of the predictions.

B. The variance in the model's mean prediction due to parameter uncertainty.

C. The total variance of the output.

D. The irreducible error.

68. What does the "Expansion" phase in MCTS involve?

A. Simulating a full game from the current node.

B. Creating new child nodes for untried actions.

C. Choosing the best child node to visit.

D. Updating the value of the parent node.

69. Why is a closed-loop policy superior to an open-loop plan in a stochastic environment?

A. It is computationally cheaper.

B. It can adapt its actions based on the actual states encountered.

C. It guarantees optimality.

D. It does not require a model.

70. The DAgger-like iterative approach for MBRL aims to make the model's training distribution...

A. ...as small as possible.

B. ...as diverse as possible by using random data.

C. ...track the state distributions encountered by the improving policies.

D. ...focus only on high-reward states.

71. How often does an agent replan in an MPC framework?

A. Only once at the beginning of an episode.

B. At every single time step.

C. Only when the model error exceeds a threshold.

D. After the episode is complete.

72. A "dual-objective" planner might balance which two goals?

A. Maximizing reward and minimizing computation time.

B. Model accuracy and policy complexity.

C. Exploration (driven by epistemic uncertainty) and safety (driven by aleatoric uncertainty).

D. Sample efficiency and asymptotic performance.

73. What is the final conclusion of the review regarding the future of intelligent systems?

A. They will rely exclusively on model-free methods.

B. They will depend on the ability to plan effectively in the face of the unknown.

C. They will require perfect, analytical models.

D. They will abandon the reinforcement learning framework.

74. The synthesis of ideas in algorithms like PETS has shown that it is possible to combine:

A. Low sample complexity and low asymptotic performance.

B. The data-frugality of MBRL with the high final performance of MFRL.

C. The simplicity of MFRL with the adaptability of MBRL.

D. High computational cost with low sample efficiency.

75. What does the term "model bias" refer to?

A. A preference for simpler models.

B. Inaccuracies in the learned model of the environment.

C. The bias term in a neural network layer.

D. A systematic error in the reward function.

76. Which of the following is a key characteristic of MFRL?

A. High sample efficiency

B. Generally higher asymptotic performance

C. High adaptability to change

D. Use of a learned world model for planning

77. Which of the following is a key characteristic of MBRL?

    A. Low sample efficiency

    B. Simpler architecture

    C. High adaptability to change

    D. Not limited by model accuracy

2 Explanatory Questions

1. Explain the core trade-off between sample efficiency and asymptotic performance when comparing Model-Free RL (MFRL) and Model-Based RL (MBRL).

2. What is ”model bias” in the context of MBRL, and why does it fundamentally limit the agent’s performance?

3. Describe the problem of ”distributional shift” in naive MBRL. Why does a model trained on a static dataset fail when used for planning? Use the ”cheetah learning to fly” example to illustrate.

4. Explain the Model Predictive Control (MPC) framework. How does it help mitigate the problem of compounding model errors?

5. What is the difference between open-loop planning and a closed-loop policy? Use the Persian anecdote of the deaf man to explain why the latter is necessary in a stochastic world.

6. Define aleatoric and epistemic uncertainty. For each type, describe its source, whether it is reducible, and what behavior it should drive in an intelligent agent.

7. How can a deep ensemble of neural networks be used to quantify both a final prediction and the epistemic uncertainty associated with it?

8. Describe the four phases of a single iteration in Monte Carlo Tree Search (MCTS): Selection, Expansion, Simulation, and Backpropagation.

9. Explain the UCT formula used in the MCTS selection phase. How do its two main components represent the exploration-exploitation trade-off?

10. What is the Cross-Entropy Method (CEM) for planning? Walk through the iterative steps of the algorithm.

11. Why is MBRL generally considered more adaptable to changes in the environment (e.g., a new goal location) than MFRL?

12. What is the DAgger (Dataset Aggregation) algorithm, and how is its core idea applied to MBRL to combat distributional shift?

13. Compare and contrast Bayesian Neural Networks (BNNs) and Deep Ensembles as methods for uncertainty quantification. What are the main pros and cons of each?

14. How does the PETS algorithm integrate an uncertainty-aware model, MPC, and trajectory optimization to achieve high performance and sample efficiency?

15. Explain how a planner can be designed to optimize a ”dual objective” that balances safety and exploration by using disentangled uncertainty signals.

16. What does it mean to say that the distinction between model-free and model-based RL is a ”spectrum” rather than a rigid binary? Use the Dyna-Q algorithm as an example.

17. Why are derivative-free optimization methods like CEM useful for planning with a learned model?
18. In MCTS, the algorithm can be viewed as solving a nested series of which simpler problem? Explain this analogy.
19. What is the ”vicious cycle” created by distributional shift in naive MBRL?
20. How does uncertainty-aware trajectory optimization (e.g., planning with the mean reward over an ensemble) implicitly avoid exploiting model flaws?
21. What is the role of the ”default policy” during the simulation (rollout) phase of MCTS? Why is a simple policy often sufficient?
22. Explain the concept of a ”value-equivalent model.” How does its objective differ from a model trained for maximum predictive accuracy?
23. Why is the backpropagation step in MCTS crucial for the algorithm's learning process?
24. If an MBRL agent is deployed in a safety-critical domain, which type of uncertainty (aleatoric or epistemic) should it be more concerned with avoiding, and why?
25. Describe the computational profile of MBRL. What makes it more demanding than MFRL?
26. How does the concept of ”planning as black-box optimization” work?
27. What is the main limitation of the simple ”random shooting” planning method?
28. Explain how the refitting step in CEM for a Gaussian distribution is a form of Maximum Likelihood Estimation (MLE).
29. Why is the ”anytime” property of MCTS useful for decision-making under time constraints?
30. Summarize the overall evolution of MBRL as described in the text, from naive approaches to modern, uncertainty-aware frameworks.

3 Answers to Multiple Choice Questions

1. B (Model-free vs. Model-based RL)
2. B (As a "black box")
3. B (To learn a policy or a value function directly.)
4. C (The true transition dynamics $p(s_{t+1}|s_t, a_t)$)
5. C (Model Learning and Planning)
6. A (A human creating a "mental map")
7. B (As a supervised learning problem)
8. C (Superior sample efficiency)
9. C (Model bias)
10. B (It learns directly from real experience and is not limited by model accuracy.)
11. B (Model-Based RL)
12. C (By re-planning with the existing dynamics model.)
13. B (Using a learned model to identify and explore uncertain regions of the state space.)
14. C (A hybrid algorithm blending model-free and model-based elements)
15. B (Trajectory optimization)
16. C (It cannot react to unexpected outcomes.)
17. B (It maps the current observed state to an action.)
18. B (Gradients of the model are unavailable or difficult to compute.)
19. C (Random Shooting)
20. B (By iteratively refining a sampling distribution to focus on high-reward regions.)
21. C (To refit the parameters of the sampling distribution.)
22. B (The sample mean and sample covariance)
23. C (Discrete action spaces)
24. B (A lightweight, randomized simulation from a node to the end of an episode.)
25. D (Pruning)
26. B (To balance exploration and exploitation during the selection phase.)
27. B (The exploitation term (mean reward))

28. B (The multi-armed bandit (MAB) problem)

29. C (Distributional shift)

30. B (The data distribution used for training the model and the distribution encountered by the planner.)

31. B (Distributional shift)

32. C (By executing only the first action of a plan and then replanning from the new state.)

33. B (Building models that explicitly quantify and reason about their own uncertainty.)

34. B (Inherent, irreducible randomness in the environment.)

35. A (Uncertainty that arises from the model's own lack of knowledge due to limited data.)

36. B (Epistemic)

37. C (Optimism and information-seeking exploration)

38. C (Inherent, unavoidable risk)

39. B (To infer a full posterior distribution over the model parameters.)

40. B (A probability distribution)

41. B (Variational Inference (VI))

42. B (They are computationally expensive and can be complex to train.)

43. B (By the variance of the predictions across the ensemble.)

44. C (They are simpler to implement and highly parallelizable.)

45. B (By averaging the reward across rollouts from an ensemble of models.)

46. C (An ensemble model, MPC, and a sampling-based trajectory optimizer (CEM))

47. B (Probabilistic Ensembles with Trajectory Sampling)

48. B (By augmenting the objective with an exploration bonus proportional to epistemic uncertainty.)

49. C (The objective should be modified to penalize it, leading to risk-averse behavior.)

50. A (The planner exploiting inaccuracies in a learned model on out-of-distribution states.)

51. B (High sample complexity)

52. C (Overcoming model bias and compounding errors)

53. B (Ensure plans rolled out in the model produce the same cumulative reward as in the real world.)

54. C (Open-loop planning)

55. B (It controls the degree of exploration.)

56. C (The visit counts and value estimates along the selection path.)

57. B (It can be run for any amount of time and will return the best action found so far.)

58. C (Trajectory optimization using calculus of variations)

59. B (The cross-entropy between the elite distribution and the sampling distribution.)

60. B (It can adapt to new tasks (e.g., new goal locations) by re-planning without new data.)

61. B (The accuracy of its learned model.)

62. B (Data collection can be expensive, time-consuming, or dangerous.)

63. C (Using the learned model to simulate outcomes and select actions.)

64. B (Epistemic is reducible, aleatoric is not.)

65. C (Bayesian inference in a BNN)

66. C (They have high computational cost due to training and running multiple models.)

67. B (The variance in the model's mean prediction due to parameter uncertainty.)

68. B (Creating new child nodes for untried actions.)

69. B (It can adapt its actions based on the actual states encountered.)

70. C (...track the state distributions encountered by the improving policies.)

71. B (At every single time step.)

72. C (Exploration (driven by epistemic uncertainty) and safety (driven by aleatoric uncertainty).)

73. B (They will depend on the ability to plan effectively in the face of the unknown.)

74. B (The data-frugality of MBRL with the high final performance of MFRL.)

75. B (Inaccuracies in the learned model of the environment.)

76. B (Generally higher asymptotic performance)

77. C (High adaptability to change)

4 Answers to Explanatory Questions

Answer to Question 1

The core trade-off between MFRL and MBRL is **sample efficiency vs. asymptotic performance**.

- **MBRL has high sample efficiency:** By learning a model of the world, an MBRL agent can generate a virtually unlimited amount of simulated experience. This allows it to learn effective policies with far fewer interactions with the real, potentially expensive or dangerous, environment.

- **MFRL has high asymptotic performance:** The performance of an MBRL agent is capped by the accuracy of its learned model (model bias). Any flaws in the model can be exploited by the planner, leading to a suboptimal policy. In contrast, MFRL learns directly from real-world data. Given enough samples, it is not constrained by a flawed model and can converge to a more optimal solution, often achieving better final performance.

Answer to Question 2

Model bias refers to the inaccuracies present in the learned model of the environment. Since the model is learned from a finite amount of data, it will never be a perfect representation of the true world dynamics.
This bias fundamentally limits the agent's performance because the planning algorithm operates on this flawed model. The planner, being an optimization process, will inadvertently find and exploit these inaccuracies. If the model erroneously predicts a high reward in a certain region of the state space, the planner will generate a policy that drives the agent to that region. This policy is optimal for the *flawed model* but will perform poorly in the *real environment*, as the expected high rewards will not materialize. Therefore, the agent's performance is ultimately capped by how accurately its model represents reality.

Answer to Question 3

Distributional shift is a fundamental problem in naive MBRL where the distribution of states and actions seen during model training is different from the distribution encountered when the model is used for planning.
A naive approach involves: 1) collecting a static dataset with an initial policy (e.g., random), 2) training a dynamics model on this data, and 3) using the model to plan an optimal policy. This fails because the new, ”optimal” policy will naturally visit different, hopefully better, states than the initial policy. The model is thus forced to make predictions for out-of-distribution inputs, where it is unreliable.
The ”cheetah learning to fly” example illustrates this perfectly. The model is trained on data of the cheetah walking. The planner, searching for high rewards, discovers a flaw in the model that suggests the cheetah can achieve immense speed by flipping over and ”flying.” This is an out-of-distribution state the model has never seen. The planner exploits this flaw, creating a policy that is nonsensical in the real world, leading to catastrophic failure. This creates a vicious cycle where the planner drives the agent further from the training data, compounding the model's errors.

Answer to Question 4

Model Predictive Control (MPC), also known as replanning, is a framework that mitigates compounding model errors by constantly re-grounding the plan in reality. Instead of creating a long-term plan and executing it blindly, MPC follows an iterative process at each time step:

1. **Plan:** From the current state observed in the real world, use the learned model to plan an optimal sequence of actions over a short, finite horizon.

2. **Execute:** Execute only the *first* action of that plan in the real environment.

3. **Observe and Repeat:** Observe the true next state that results from that action. Discard the rest of the old plan and repeat the entire process from this new, real state.

By replanning at every step, MPC prevents one-step prediction errors from accumulating over a long trajectory. The plan is only ever based on the most recent, true state, making the system far more robust to the inevitable inaccuracies of the learned model.

Answer to Question 5

- **Open-loop planning** involves determining a fixed sequence of actions *before* execution. The agent commits to this entire sequence regardless of what happens in the environment.

- **A closed-loop policy** is a reactive strategy or function that maps the current observed state to an action. It allows the agent to dynamically adapt its behavior based on the actual trajectory it experiences.

The Persian anecdote of the deaf man perfectly illustrates the failure of open-loop planning. The man rehearses a fixed sequence of responses (”Thank God!”, ”May it be good for you!”). His plan is open-loop. When the sick friend’s replies (”I am dying!”, ”I have eaten poison!”) deviate from his expectations, his pre-planned responses become tragically inappropriate. A closed-loop policy would have allowed him to adapt his response based on the friend’s actual state (his words), demonstrating why such reactive strategies are essential in a stochastic or unpredictable world.

Answer to Question 6

- **Aleatoric Uncertainty:**
  - **Source:** Inherent, irreducible randomness in the data-generating process or environment itself (e.g., sensor noise, stochastic dynamics).
  - **Reducibility:** It **cannot** be reduced by collecting more data.
  - **Implication:** It signals unavoidable **risk**. An intelligent agent should be cautious or risk-averse in the face of high aleatoric uncertainty.

- **Epistemic Uncertainty:**
  - **Source:** The model’s own lack of knowledge, typically due to being trained on a finite or limited dataset.
  - **Reducibility:** It **can** be reduced by collecting more relevant data in the uncertain regions.
  - **Implication:** It signals **ignorance**. An intelligent agent should be optimistic and treat it as a signal for information-seeking **exploration**.

Answer to Question 7

A deep ensemble consists of multiple (N) individual neural networks, each trained on the same dataset but with different random initializations (and optionally, different bootstrapped data samples). To get a prediction and its uncertainty for a new input:

1. The input is passed through all N models in the ensemble.
2. This yields N different predictions.
3. The **final prediction** is taken as the **mean** of these N individual predictions. This averaging process tends to produce a more robust and accurate prediction than a single model.
4. The **epistemic uncertainty** is quantified by the **variance** of the N predictions. If the models, having been trained from different starting points, all agree on the output, the variance will be low, indicating low uncertainty. If their predictions diverge significantly, the variance will be high, indicating that the models are uncertain because they are extrapolating into a region with little data.

Answer to Question 8

A single iteration of MCTS consists of four phases:

1. **Selection:** Starting from the root node (the current state), the algorithm traverses the existing search tree. At each node, it uses a tree policy (like UCT) to select a child node that optimally balances exploiting known good paths and exploring less-visited ones. This continues until a leaf node (a node that is not fully expanded) is reached.
2. **Expansion:** If the selected leaf node is not a terminal state, the tree is expanded by creating one or more new child nodes corresponding to valid, untried actions from that state.
3. **Simulation (Rollout):** From one of these new nodes, a simulation is run. The algorithm follows a simple, fast ”default policy” (e.g., random actions) until a terminal state is reached. The total reward from this simulated trajectory is recorded.
4. **Backpropagation:** The outcome (total reward) of the simulation is propagated back up the tree along the path of nodes visited during the selection phase. The statistics (visit count N and value estimate Q) of each node-action pair on this path are updated with the new result.

Answer to Question 9

The UCT (Upper Confidence Bound 1 for Trees) formula is:

$$\text{Score}(s_{t+1}) = \underbrace{\frac{Q(s_{t+1})}{N(s_{t+1})}}_{\text{Exploitation}} + \underbrace{C\sqrt{\frac{\ln N(s_t)}{N(s_{t+1})}}}_{\text{Exploration}}$$

It balances the exploration-exploitation trade-off through its two main components:

1. **Exploitation Term (Q/N):** This is the current average reward for the child node. It favors selecting actions that have historically led to high rewards. This is the "exploit" part—go with what you know works well.

2. **Exploration Term (the square root term):** This term provides an "uncertainty bonus." It is large for child nodes that have been visited infrequently ($N(s_{t+1})$ is small) relative to their parent ($N(s_t)$). This encourages the algorithm to try actions whose true values are still highly uncertain, preventing it from prematurely settling on a suboptimal choice. The constant C controls how much weight is given to exploration.

Answer to Question 10

The Cross-Entropy Method (CEM) is a derivative-free, evolutionary-style optimization algorithm used for planning. It iteratively refines a probability distribution over action sequences to focus on high-reward regions. The steps are:

1. **Initialization:** Define a parameterized probability distribution over action sequences, typically a multivariate Gaussian with a mean μ and covariance Σ.

2. **Sample:** Draw a population of N candidate action sequences from the current distribution.

3. **Evaluate:** Use the dynamics model to simulate each sequence and calculate its total cumulative reward.

4. **Select Elites:** Identify the top-performing fraction (e.g., top 10-20%) of the action sequences. These are the "elites."

5. **Refit Distribution:** Update the parameters of the sampling distribution (μ and Σ) to better fit the elite samples. For a Gaussian, this is done by calculating the sample mean and covariance of the elite set.

6. **Iterate:** Repeat the sample-evaluate-refit loop for a fixed number of iterations or until the distribution converges.

Answer to Question 11

MBRL is more adaptable because it decouples the model of the world's dynamics from the task-specific goal.

- An **MBRL agent** learns a general-purpose model of how the environment works ($p(s'|s, a)$). If the task changes—for example, the goal location moves, which only alters the reward function $r(s, a)$—the agent does not need to relearn the world dynamics. It can simply use its existing model and **re-plan** with the new reward function to find an effective policy for the new task, often without collecting any new environmental data.
- An **MFRL agent**, by contrast, learns a policy or value function that is implicitly tied to a specific task and reward structure. If the task changes, the entire learned policy is likely invalid, and the agent would need substantial retraining from scratch through many new interactions with the environment.

Answer to Question 12

DAgger (Dataset Aggregation) is an algorithm from imitation learning that addresses distributional shift. Its core idea is to iteratively collect data using the current best policy and aggregate it with previous data to retrain the policy.
This idea is applied to MBRL to combat distributional shift as follows:

1. **Initialize:** Start with a dataset $\mathcal{D}$ collected with a base policy π_0.
2. **Iterate:**
   (a) **Learn Model:** Learn the dynamics model f on the current dataset $\mathcal{D}$.
   (b) **Derive Policy:** Use the model f to derive an improved policy π_f.
   (c) **Collect New Data:** Use this new policy π_f to interact with the *real* environment and collect new transitions.
   (d) **Aggregate:** Add these new transitions to the dataset: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\text{new}}$.

This "closes the loop" by forcing the model's training distribution to track the state distributions that are actually encountered by the improving policies, ensuring the model remains accurate in the regions where the planner operates.

Answer to Question 13

Bayesian Neural Networks (BNNs) and **Deep Ensembles** are two methods for uncertainty quantification in deep learning.

- **BNNs:**
  - **Method:** Learns a probability distribution (e.g., a Gaussian with mean and variance) for each weight in the network, representing a distribution over models.
  - **Pros:** Principled and theoretically grounded in Bayesian statistics. Can provide a form of automatic regularization.
  - **Cons:** Computationally expensive (doubles the number of parameters), training can be complex and unstable, and performance is sensitive to the choice of prior and the quality of the variational approximation.
- **Deep Ensembles:**
  - **Method:** Trains multiple independent neural networks with different random initializations. Uncertainty is the variance in predictions across the ensemble members.
  - **Pros:** Remarkably simple to implement, highly parallelizable, and empirically shown to produce high-quality, well-calibrated uncertainty estimates that often match or exceed BNNs.
  - **Cons:** High computational cost (training and inference time scales linearly with ensemble size), and it is considered a more heuristic or "crude" approximation to a true Bayesian posterior compared to BNNs.

In practice, Deep Ensembles are often preferred for their simplicity and strong empirical performance.

Answer to Question 14

PETS (Probabilistic Ensembles with Trajectory Sampling) is a high-performance MBRL algorithm that successfully integrates three key concepts:

1. **Uncertainty-Aware Model:** It learns a probabilistic dynamics model using a **deep ensemble** of neural networks. This allows it to quantify its epistemic uncertainty by measuring the disagreement among the models in the ensemble.

2. **Model Predictive Control (MPC):** It operates within an MPC framework. At each step, it plans a sequence of actions but only executes the first one, then observes the real outcome and replans. This makes the system robust to model errors by preventing them from compounding.

3. **Uncertainty-Aware Trajectory Optimization:** It uses a sampling-based planner (specifically, CEM) to find the best action sequence. Crucially, the objective for CEM is to maximize the **average reward across the ensemble of models**. This implicitly penalizes plans that lead to high-disagreement (high-uncertainty) states, preventing the planner from exploiting the flaws of any single model.

This combination allows PETS to be highly sample-efficient (from MBRL) while achieving the high asymptotic performance of top MFRL methods.

Answer to Question 15

A planner can optimize a ”dual objective” by leveraging the disentangled signals of epistemic and aleatoric uncertainty to balance safety and exploration:

- **Balancing for Exploration:** To encourage exploration, the planner’s objective function can be augmented with a bonus proportional to epistemic uncertainty. For example:

$$J(A) = (\text{mean reward}) + \lambda \times (\text{epistemic uncertainty})$$

  This modification explicitly incentivizes the agent to take actions that lead to novel states where its model is uncertain, thereby gathering the most informative data to improve its world model. This is ”optimism in the face of uncertainty.”

- **Balancing for Safety:** For safety-critical applications, the objective can be modified to penalize aleatoric uncertainty. For example:

$$J(A) = (\text{mean reward}) - \gamma \times (\text{aleatoric uncertainty})$$

  This creates a risk-averse agent that actively avoids regions of the state space that are inherently unpredictable and risky, even if they offer potentially high rewards. This is ”pessimism in the face of risk.”

Answer to Question 16

The distinction is a ”spectrum” because many modern, effective algorithms are not purely model-free or model-based but are hybrids that blend elements of both paradigms. The design choice is not ”which one to use?” but rather ”how can a model be best leveraged to improve learning?”
The **Dyna-Q** algorithm is a classic example.

- It has a **model-based component**: it learns a model of the environment from real interactions.
- It has a **model-free component**: its primary learning algorithm is a model-free method like Q-learning.

Instead of using the model for complex, multi-step planning, Dyna-Q uses it simply as a **data augmentation tool**. It generates simulated experiences (state, action, next_state, reward) from the model and adds them to the replay buffer of the Q-learning algorithm. This allows the model-free learner to be updated with both real and simulated data, dramatically improving its sample efficiency while retaining the core model-free learning mechanism. This places it squarely between the two extremes.

Answer to Question 17

Derivative-free optimization methods are useful for planning with a learned model because the model is often a complex, non-differentiable ”black box.” This can happen for several reasons:

1. **Complex Architecture:** The learned model is typically a deep neural network, and while it's often differentiable, computing gradients through many unrolled time steps can be computationally expensive and lead to vanishing or exploding gradients.
2. **Black-Box Simulators:** In some cases, the ”model” might be an existing simulator (e.g., for physics or economics) for which we do not have access to the internal workings or gradients.
3. **Simplicity and Generality:** Derivative-free methods treat the planning problem as finding an action sequence that maximizes a ”score” (the cumulative reward from the model). They don't need to know *how* the model works, only what output it produces for a given input. This makes them very general and easy to apply to any system that can be simulated.

Methods like CEM and Random Shooting simply ”guess and check” action sequences, making them applicable even when gradient information is unavailable or impractical to compute.

Answer to Question 18

At each node in the search tree, MCTS can be viewed as solving a local **Multi-Armed Bandit (MAB)** problem.
The analogy is as follows:

- **The Bandit Machine:** The current node in the tree.
- **The Arms:** The possible actions (leading to child nodes) that can be taken from the current node.
- **Pulling an Arm:** Selecting an action and traversing to the corresponding child node.
- **The Reward for Pulling an Arm:** This is the crucial part. The ”reward” is not immediate but is estimated by the entire complex machinery of the MCTS process that follows: the simulation (rollout) from the new node and the backpropagation of its outcome.

The UCT formula is a sophisticated policy for solving this MAB problem at each node, deciding which ”arm” to pull to best balance exploring the different actions and exploiting the ones that have proven valuable. MCTS thus cleverly decomposes a complex, long-horizon sequential decision problem into a nested series of one-step MAB problems.

Answer to Question 19

The ”vicious cycle” of distributional shift in naive MBRL is a cascading failure mode:

1. **Initial Model Error:** The model is trained on data from an initial policy and is inevitably imperfect, especially in regions of the state-space it hasn't seen.
2. **Planner Exploitation:** The planning algorithm, being an optimizer, actively seeks out and exploits any inaccuracies in the model that promise high rewards. It finds ”fantasy” trajectories that are good in the flawed model but not in reality.
3. **Out-of-Distribution States:** These fantasy trajectories lead the agent into out-of-distribution (OOD) states, far from its training data.
4. **Compounding Errors:** In these OOD regions, the model's predictions become even more unreliable, and its errors compound. The planner, still trusting the flawed model, continues to push the agent further into these nonsensical regimes.

This cycle leads to a policy that is completely detached from reality and performs very poorly in the real world.

Answer to Question 20

Uncertainty-aware trajectory optimization avoids exploiting model flaws by averaging out the "fantasies" of individual models. The process works as follows:

1. An action sequence is proposed by the planner (e.g., CEM).
2. This sequence is simulated across an entire ensemble of N different learned models.
3. If the sequence leads to a region of the state space where the models are certain (low epistemic uncertainty), all N models will agree on the outcome, and the predicted reward will be consistent.
4. However, if the sequence leads to an out-of-distribution region where a single model might have a "fantasy" high-reward prediction, the other models in the ensemble are unlikely to have the exact same flaw. Their predictions will diverge, resulting in a high variance of predicted returns.
5. The planner's objective is to optimize the **mean** return across the ensemble. The high variance from the diverging "fantasy" trajectories will result in a less appealing mean return compared to a plan where all models agree on a moderately good outcome.

Therefore, the planner naturally shies away from these uncertain, high-disagreement regions and prefers robust plans, implicitly avoiding the exploitation of any single model's flaws.

Answer to Question 21

The **default policy** is a simple, fast policy used during the simulation (rollout) phase of MCTS to play out an episode from a newly expanded leaf node to a terminal state.
A simple policy (e.g., choosing actions uniformly at random) is often sufficient for two main reasons:

1. **Speed is Critical:** MCTS's power comes from running thousands or millions of simulations. The rollout needs to be as fast as possible. A complex policy would make each simulation too slow, limiting the number of iterations and thus the quality of the search.
2. **Averaging Effect:** The goal of the rollout is not to find the optimal path, but to get a quick, noisy estimate of the value of the leaf node. While a single random rollout might be a poor estimate, MCTS performs many of them. The law of large numbers ensures that the *average* of these many noisy estimates provides a reasonably good signal to guide the more precise UCT-based search in the upper parts of the tree. The main search effort is focused near the root, not in the rollouts.

Answer to Question 22

A **value-equivalent model** is a type of learned model whose training objective is different from a standard dynamics model.

- A **standard dynamics model** is trained for maximum predictive accuracy. Its goal is to minimize the error between its predicted next state, $\hat{s}'$, and the true next state, s', (e.g., by minimizing $||\hat{s}' - s'||^2$).

- A **value-equivalent model**, in contrast, is trained with the objective that plans (or "rollouts") executed within the model should produce the same *cumulative reward* (or value) as equivalent trajectories in the real world. It doesn't need to predict the states themselves with perfect pixel-for-pixel accuracy, as long as the *value* of those predicted states is correct.

This reframes the goal from "predict the future accurately" to "predict the *value* of the future accurately," which can be a more direct and sometimes easier objective for the purpose of planning.

Answer to Question 23

The backpropagation step is crucial because it is the mechanism through which the tree **learns** from the simulations. It connects the results of the random rollouts to the strategic decisions made higher up in the tree.

When a simulation is completed, its outcome (the total reward) is a new piece of information about the value of the leaf node from which the simulation started. The backpropagation step takes this new information and updates the statistics (the visit count N and the value estimate Q) of *every node-action pair on the path that led to that leaf node.*

This update is vital because it refines the value estimates that the UCT formula relies on for the next selection phase. Actions that lead to rollouts with high rewards will have their Q-values increased, making them more likely to be selected in the future (exploitation). Actions that have been tried will have their N-values increased, which reduces their exploration bonus, allowing other, less-tried actions to be explored. Without backpropagation, the results of the simulations would be lost, and the tree would never improve its estimates.

Answer to Question 24

In a safety-critical domain, the agent should be more concerned with avoiding **aleatoric uncertainty**.
Here's why:

- **Epistemic uncertainty** represents the agent's own ignorance. While it can lead to poor decisions, it is fundamentally *reducible*. The agent can perform safe exploratory actions to gather data and reduce this uncertainty over time. It is a problem of ”not knowing.”

- **Aleatoric uncertainty** represents inherent, irreducible randomness in the environment. It signals that a region of the state space is fundamentally unpredictable and risky, no matter how much the agent knows about it. This is a problem of ”inherent danger.”

For safety, the primary goal is to avoid catastrophic failures. Therefore, the agent must prioritize avoiding states and actions where the outcome is uncontrollably random and potentially dangerous (high aleatoric uncertainty), even if those states could potentially offer high rewards.

Answer to Question 25

The computational profile of MBRL is generally more complex and demanding than MFRL due to two main components:

1. **Model Learning and Maintenance:** MBRL involves the overhead of training and storing an environment model, which is often a large neural network or an ensemble of them. This model needs to be periodically retrained or updated as new data comes in, which adds to the computational load.

2. **Planning:** The planning phase itself can be very computationally intensive, especially at decision time. Algorithms like MCTS or CEM require running many simulations inside the learned model to decide on a single action. This ”thinking time” can be significant, whereas a trained MFRL policy can often make a decision with a single, fast forward pass through a neural network.

In short, MBRL shifts the computational burden from requiring a vast number of real-world samples (like MFRL) to requiring significant computation for model training and planning.

Answer to Question 26

”Planning as black-box optimization” is a framework for decision-making that treats the learned dynamics model as a function whose internal workings are unknown or ignored. The goal is to find an input (an action sequence) that produces the best possible output (the highest cumulative reward).
The process works as follows:

1. The ”black box” is the function that takes a starting state and a sequence of actions and, by using the learned model, simulates the trajectory and outputs a single number: the total reward.
2. The planning algorithm’s job is to search for the optimal action sequence without needing to know the gradients or the structure of the model.
3. It does this by generating candidate action sequences, ”feeding” them to the black-box model to get their scores, and then using those scores to intelligently generate the next set of candidates.

Methods like Random Shooting and the Cross-Entropy Method (CEM) are examples of this paradigm. They are powerful because they can be applied to any model that can be simulated, regardless of its complexity or differentiability.

Answer to Question 27

The main limitation of the simple ”random shooting” planning method is its poor scalability, a direct result of the **curse of dimensionality**.
The search space for an action sequence is the action space dimension raised to the power of the planning horizon ($|\mathcal{A}|^H$). This space grows exponentially with the length of the plan (H) and the size of the action space. Randomly sampling (or ”shooting”) candidate sequences becomes incredibly inefficient in a high-dimensional space, as the vast majority of random sequences will be nonsensical and have very low rewards. The probability of randomly hitting a high-performing sequence becomes vanishingly small, making the method impractical for all but the simplest problems with short horizons and small action spaces.

Answer to Question 28

In the Cross-Entropy Method (CEM), the refitting step updates the parameters of the sampling distribution (e.g., mean μ and covariance Σ for a Gaussian) to best fit the ”elite” samples—the action sequences that produced the highest rewards.
This is a form of Maximum Likelihood Estimation (MLE) because the goal is to find the distribution parameters that **maximize the probability (or likelihood) of observing the elite samples**. For a Gaussian distribution, the MLE parameters that best explain a given set of data points are simply the sample mean and sample covariance of that data. Therefore, by calculating the mean and covariance of the elite set and setting them as the new parameters for the sampling distribution, CEM is performing MLE to shift its search towards the promising region of the action space identified by the elites.

Answer to Question 29

The ”anytime” property of MCTS means that the algorithm can be interrupted at **any time** and it will still be able to provide the best action found so far.
This is extremely useful for decision-making under time constraints (e.g., in real-time games or robotics) because:

- The algorithm's performance gracefully degrades or improves with the amount of computation time allocated. If there is a lot of time before a decision is needed, MCTS can run for many iterations and find a very strong move.
- If a decision must be made immediately, MCTS can be stopped after just a few iterations and will return the action that currently has the highest visit count or value, which is still a reasonable, informed choice.

Unlike some algorithms that must run to completion to produce any result, MCTS continuously refines its answer, making it flexible and robust in dynamic environments where thinking time is a limited resource.

Answer to Question 30

The evolution of MBRL described in the text follows a path from brittle, naive methods to robust, uncertainty-aware frameworks:

1. **Naive MBRL:** The initial paradigm was a simple ”learn, then plan” approach. This was plagued by **distributional shift** and **compounding model errors**, making it impractical as planners would exploit model flaws.
2. **Mitigation Strategies:** The first major improvements came from techniques to make the process more robust. **Iterative data aggregation** (like DAgger) helped align the training and planning distributions, while **Model Predictive Control (MPC)** prevented error accumulation by constantly replanning with real-world feedback.
3. **The Uncertainty Breakthrough:** The true leap forward was the explicit incorporation of uncertainty. Instead of just trying to learn the most accurate model, the focus shifted to building models (like **BNNs** or **Deep Ensembles**) that could quantify their own ignorance (**epistemic uncertainty**) and the world's randomness (**aleatoric uncertainty**).
4. **Modern, Uncertainty-Aware Frameworks:** This new capability enabled intelligent planning. Planners could now optimize objectives that averaged over an ensemble of futures to avoid exploiting model flaws. This culminated in algorithms like **PETS**, which synthesize these ideas—ensembles, MPC, and uncertainty-aware planning—to achieve both the sample efficiency of MBRL and the high asymptotic performance of MFRL. The final stage is using disentangled uncertainty for dual-objective planning (balancing exploration and safety).