Reinforcement Learning Lab

Module: Optimal Policies Under Markov Decision Processes

Author: Charalampos (Babis) Giannakis
Affiliation: MSc Business Analytics – Computational Intelligence, VU Amsterdam
Version: v1.0 **Project ID:** RL-2025-02
Date: November 30, 2025
Contact: github.com/CharalamposGiannakis

We start from structure. Complexity follows.

Introduction

We study a discrete-time Markov decision process describing the maintenance of two deteriorating components. Each component evolves through ten states, and the controller may apply corrective and, depending on the setting, preventive repair. The objective is to maximise the long-run average reward. The report first specifies the MDP (part (a)), then computes the long-run reward without preventive repair using simulation, the stationary distribution, and the Poisson equation (part (b)). Finally, optimal policies are derived when preventive repair is only possible after a failure (part (c)) and when it is always allowed (part (d)). All computations were implemented in Python and the report prepared in LaTeX.

a) Components of the MDP

We model the system as an MDP. Each component has deterioration levels 1 (new) to 10 (failed), so the state space is

$$S = \{1, \dots, 10\}^2,$$

consisting of 100 states (i, j).

Actions. When both components are operational ($i, j < 10$), actions are

$$\{\text{operate},\ \text{repair}_1,\ \text{repair}_2,\ \text{repair}_{12}\}.$$

If one component has failed, corrective repair of that component is required, optionally combined with preventive repair of the healthy one. If both have failed, both must be repaired.

Transitions. Under *operate*, each component deteriorates independently with probability 0.1:

$$\Pr(i \to i') = \begin{cases} 0.9, & i' = i, \\ 0.1, & i' = \min\{i+1, 10\}. \end{cases}$$

Repair actions take one time step, reset repaired components to 1, and keep unrepaired components fixed.

Rewards. Operating when both components are working yields reward 1. Preventive and corrective repair incur costs 5 and 25 per component respectively, giving

$$r((i,j), a) = 1 \text{ if } a = \text{operate},\ i, j < 10, \qquad r = -5n_p - 25n_c \text{ otherwise.}$$

b) Long-run reward

1. Simulation

Under the no-preventive policy (operate if both work, otherwise correct all failed components), a trajectory of $T = 10^6$ steps was simulated. The estimator

$$\hat{g}_T = \frac{1}{T} \sum_{t=1}^{T} R_t$$

stabilised at $\hat{g}_{10^6} = 0.4371$.

2. Stationary distribution

The chain under the fixed policy is irreducible (every state can reach every other because you can always repair to state (1,1) and deteriorate again), and has self-loops (e.g. staying in the same state under operate), so it is aperiodic. Therefore the stationary distribution is unique and the forward recursion converges to it. We iterated $\pi^{(k+1)} = \pi^{(k)}P$ until convergence. The stationary distribution yielded $g = \sum_{s\in S} \pi(s)\, r(s) = 0.4348,$ consistent with the simulation.

3. Poisson equation

Relative value iteration was applied to the Poisson equation $g + V(s) = r(s) + \sum_{s'} P(s, s')V(s')$. Normalising $V(s_0) = 0$ at each iteration and iterating to convergence gave $g = 0.4348$, again matching the previous values.

Conclusion: The stationary distribution and Poisson equation approaches both rely on the Markov chain being unichain, which holds here because the system can always be brought back to (1,1) by repairs.

c) Optimal policy with one-sided preventive repair

Here preventive repair is allowed only when the other component has failed. Relative value iteration was applied to evaluate, in each failure configuration, whether repairing only the failed component or both is optimal. The long-run optimal reward was

$$g^* = 0.5857.$$

Preventive repair is only chosen when the healthy component is already near failure; otherwise corrective-only repair is optimal. The appendix plot shows a narrow band of states where "corrective + preventive" is optimal.

d) Optimal policy with full preventive repair

When preventive repair is allowed in any state, the controller can renew components proactively. Solving the average-reward MDP via relative value iteration yielded

$$g^* = 0.8542,$$

a substantial improvement over parts (b) and (c). The optimal policy applies preventive repair whenever either component reaches a high deterioration level, forming a cross-shaped preventive region near the failure boundary (see appendix). This reduces the probability of costly corrective failures.

Note: The optimal policies in (c) and (d) exhibit a threshold structure in the deterioration levels: preventive repair is only chosen once a component is above a certain level, which is typical for deteriorating-system control problems.

Conclusion

All three methods for the no-preventive case produced a long-run reward of about 0.435, confirming correctness of both the model and implementation. Allowing preventive repair after a failure increased the reward to roughly 0.586, improving reliability without excessive preventive cost. Allowing preventive repair at all times further increased the reward to about 0.854, and the optimal policy became strongly preventive near the failure boundary. These results align with intuition: increased flexibility in preventive maintenance leads to fewer corrective interventions and a higher long-run reward.

Appendix: Policy Tables and Plots

This appendix presents the optimal policies for parts (c) and (d). A value of 1 indicates that the selected action includes preventive repair.

Part (c): One-sided preventive repair

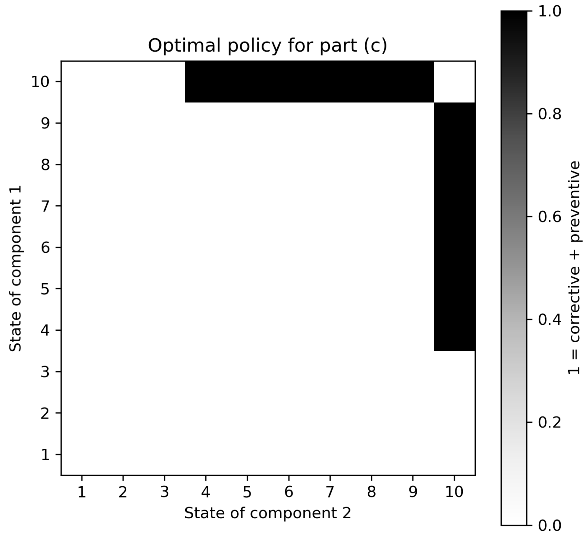

Figure (c): Optimal policy plot

Policy Table (c)

0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	1	0

Part (d): Full preventive repair

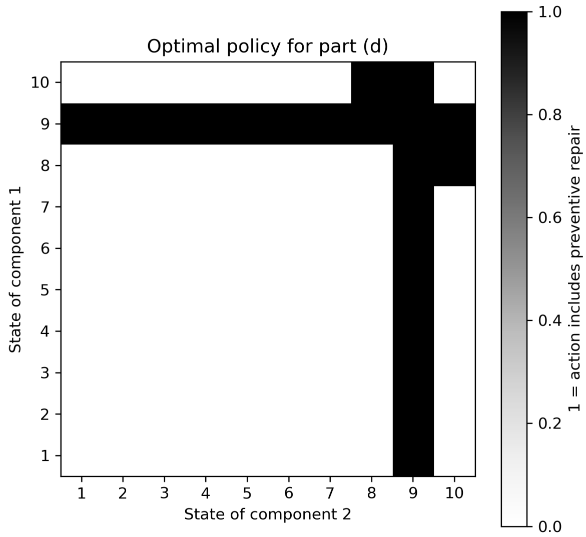

Figure (d): Optimal policy plot

Policy Table (d)

0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	1	1