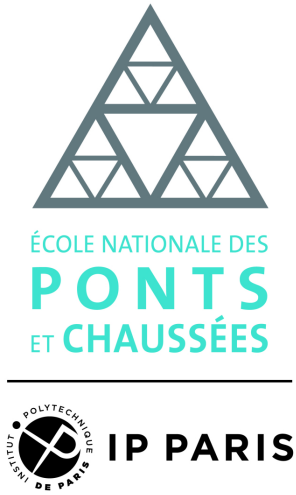
Ecole Nationale des Ponts et Chaussées



Rapport Final

Enseignant : Vincent LECLERE

Dynamic Programming Project

Hamza Ferssiwi et Mouad Jbili
Département IMI
02 Mai 2025

Contents

1 Maintenance Problem

(a) Justification for Expected Revenue Maximization

Maximizing expected revenue is appropriate due to:

- **Stochastic transitions**: Machine degradation follows probabilistic rules
- **Long-term planning**: Decisions affect future states and rewards
- **Risk neutrality**: Company seeks average optimal performance over many machines
- **Trade-off analysis**: Balances immediate costs vs future earnings systematically

(b) Controlled Markov Chain Formulation

State space:

$$\mathcal{S} = \{\text{N (New)}, \text{G (Good)}, \text{O (Old)}, \text{B (Broken)}\}$$

Action space:

- DN: Do Nothing (cost 0)
- M: Maintain (cost 10, available in G/O)
- R: Repair (cost 15/30/50 in G/O/B)
- Rp: Replace (cost 70, available in all states)

Transition dynamics (partial):

State	Action	Next State	Probability
G	DN	G	0.3
G	DN	O	0.7
G	M	G	0.8
G	M	O	0.2
O	DN	O	0.5
O	DN	B	0.5
O	M	O	0.9
O	M	B	0.1

Rewards:

$$r(s) = \begin{cases} 30 & s = \text{N} \\ 20 & s = \text{G} \\ 10 & s = \text{O} \\ 0 & s = \text{B} \end{cases}$$

(c) Dynamic Programming Solution

Value function definition:

$$V(t,s) = \max_{a \in \mathcal{A}(s)} \underbrace{[r(s) - c(a,s)]}_{\text{Immediate net reward}} + \underbrace{\sum_{s'} P(s'|s,a)V(t+1,s')}_{\text{Expected future value}}$$

Where:

- $t \in \{1, \ldots, 12\}$: Current month in planning horizon
- $s \in \mathcal{S}$: Current machine state (N/G/O/B)
- $a \in \mathcal{A}(s)$: Available actions (DN/M/R/Rp)
- $r(s)$: Immediate revenue from state s:

$$r(s) = \begin{cases} 30 & s = \text{N} \\ 20 & s = \text{G} \\ 10 & s = \text{O} \\ 0 & s = \text{B} \end{cases}$$

- $c(a,s)$: Action cost:

$$c(a,s) = \begin{cases} 0 & a = \text{DN} \\ 10 & a = \text{M} \\ 15 & a = \text{R}, s = \text{G} \\ 30 & a = \text{R}, s = \text{O} \\ 50 & a = \text{R}, s = \text{B} \\ 70 & a = \text{Rp} \end{cases}$$

- $P(s'|s,a)$: Transition probabilities from state s to s' under action a
- $V(t+1,s')$: Future value function at next time step

Algorithm:

1. Initialize $V(13,s) = 0 \ \forall s \in \mathcal{S}$
2. For $t = 12$ down to 1:
    - For each state $s \in \mathcal{S}$:
    - Compute expected values for all feasible actions
    - Choose action maximizing net reward

Example calculation (t=12, s=G):

$$\begin{aligned} Q(\text{DN}) &= 20 - 0 + 0 = 20 \\ Q(\text{M}) &= 20 - 10 + 0 = 10 \\ Q(\text{R}) &= 20 - 15 + 0 = 5 \\ V(12, G) &= \max\{20, 10, 5\} = 20 \Rightarrow \text{Optimal: DN} \end{aligned}$$

Key insights:

- Preventive maintenance valuable in early periods
- Replacement rarely optimal except for broken machines
- Repair becomes preferable as machines age

The execution of the former algorithm enables us to get ahold of the optimal expected value at the initial state **N** in the first month:

$$\boxed{V(1, \text{N}) = \texttt{110.82}}$$

State	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12
New	DN	DN	DN	DN	DN	DN	DN	DN	DN	DN	DN	–
Good	M	M	M	M	M	M	M	R	M	DN	DN	–
Old	R	R	R	R	R	R	R	R	R	DN	DN	–
Broken	Rp	Rp	Rp	Rp	Rp	Rp	Rp	Rp	DN	DN	DN	–

Legend:

- DN = Do nothing
- M = Maintain
- R = Repair
- Rp = Replace
- – = Invalid action index (terminal month, no future)

Analysis:

- **State "New"**: The optimal action is to do nothing throughout the entire period. This may be expected, as a new machine does not degrade immediately and provides the highest revenue (30). Any intervention (maintenance, repair, replacement) would incur unnecessary costs.

- **State "Good"**: The strategy recommends regular preventive maintenance until month 7, followed by a repair in month 8, and inaction in the final months. This suggests that maintenance is economically justified early on to delay degradation to the "Old" state. Toward the end of the horizon, the benefit of maintenance no longer outweighs its cost.
- **State "Old"**: The machine is repaired continuously until month 9, then left untouched. Repairing an old machine (cost 30) remains profitable as long as there is enough time left to benefit from restoring it to a "Good" state. Late in the horizon, repairs are no longer justified.
- **State "Broken"**: The policy is aggressive, recommending immediate replacement during the first 8 months, then switching to inaction. This reflects the fact that a broken machine brings in no revenue, and replacement is the only way to resume profitability. However, at the end of the horizon, the high replacement cost is no longer recoverable.

2 Stock Management Problems

2.1 Simple Stock Problem

(a) Stochastic Control Formulation

- **State**: Stock level $x_t \in \{1, \dots, 20\}$
- **Control**: Order quantity $u_t \in \{0, \dots, 5\}$ with $x_t + u_t \leq 20$ so that : $u_t \in \{0, \dots, min(5, 20 - x_t)\}$
- **Cost**:

$$c(t, x_t, u_t, d_t) = \underbrace{u_t}_{\text{Order}} + \underbrace{0.1(x_t + u_t - s_t)}_{\text{Holding}} - \underbrace{3s_t}_{\text{Revenue}}$$

  where $s_t = \min(d_t, x_t + u_t)$
- **Dynamics**:

$$x_{t+1} = \max(0, x_t + u_t - d_t), \quad d_t \sim \mathrm{B}(p_t, 10)$$

(b) Bellman Equation

$$V(t, x) = \min_{u \in \mathcal{U}(x)} [E_{d_t}[c(t, x, u) + V(t + 1, x')]]$$

with terminal condition $V(15, x) = 0$. Here $x' = \max(0, x + u - d_t)$.

(c) Policy and Simulator

- **Policy**: Function $\pi : N \times \{0, \ldots, 20\} \to \{0, \ldots, 5\}$ that returns the number of units to order such that the expected total cost is minimized over the planning horizon.
- **Simulator** key steps:
  1. Initialize $x = 10$
  2. For each day $t = 1, \ldots, 14$:
     - Get order $u = \pi(t, x)$
     - Sample demand $d_t \sim \mathrm{B}(p_t, 10)$
     - Compute costs and update stock
  3. Return average cost with 95% CI

(d) Policy simulation

We tested a simple heuristic policy where the shop orders 5 products each day, as long as the stock constraint $x_t + u_t \leq 20$ is respected. This policy can be implemented as:

$$\pi(t, x) = \min(5,\ 20 - x)$$

We evaluated the expected total cost over the 14-day horizon using Monte Carlo simulation with $N = 10{,}000$ independent runs. The stochastic demand $d_t \sim \text{Binomial}(10, p_t)$ varies across time according to the predefined vector:

$$p = [0.2,\ 0.2,\ 0.4,\ 0.4,\ 0.7,\ 0.7,\ 0.2,\ 0.2,\ 0.8,\ 0.8,\ 0.5,\ 0.5,\ 0.2,\ 0.2]$$

The results of the simulation are:

- **Estimated expected cost:** $\boxed{98.17}$
- **95% confidence interval:** $\boxed{[97.88,\ 98.47]}$

The constant-order policy used in the simulation is a naive but intuitive heuristic: always order the maximum amount (5 units) if stock permits. This ensures that:

- The shop rarely misses out on sales during high demand days, maximizing revenue.
- However, the policy incurs high storage costs during low-demand periods, particularly near the end of the planning horizon.
- It does not exploit the time-varying demand profile p_t, which contains useful predictive information.
- It also ignores the fact that excess stock has no terminal value after day 14, leading to waste.

(e) Dynamic Programming

$$V(15, x) = 0 \quad \forall x$$
$$V(t, x) = \min_u \sum_{d=0}^{10} \binom{10}{d} p_t^d (1 - p_t)^{10-d} \times \left[u + 0.1(S - d^+) - 3d^+ + V(t + 1, S - d)\right]$$

where $S = x + u$, $d^+ = \min(d, S)$

We implemented this dynamic programming algorithm via backward induction, precomputing binomial distributions for all time steps. At each time t and stock level x, we evaluated all admissible control values $u \in \{0, \ldots, \min(5,\ 20 - x)\}$ and selected the one minimizing the expected total cost.

The optimal value function $V(t, x)$ and policy matrix $\pi(t, x)$ were stored. Furthermore, the computed value at time $t = 1$ and stock $x = 10$ is:

$$\boxed{V(1, 10) = 116.43}$$

(f) Simulation of the Optimal Policy

To validate the performance of the optimal policy obtained from dynamic programming, we used the same Monte Carlo simulator described in part (c). The simulator was run with $N = 10{,}000$ independent samples of the stochastic demand.

The following results were obtained:

- **Estimated expected cost under optimal policy:** $\boxed{116.43}$
- **95% confidence interval:** $\boxed{[116.21,\ 116.65]}$

The simulated result is consistent with the value function $V(1, 10)$, confirming the correctness of the implementation and optimality of the computed policy. The narrow confidence interval demonstrates statistical reliability.

The dynamic programming solution effectively adapts the ordering decision based on:

- Current stock level,
- Remaining time horizon,
- Time-varying demand probabilities.

This contrasts with the fixed greedy policy and achieves improved performance by reducing overstocking during low-demand periods and ensuring availability during peaks.

2.2 Advanced Stock Problem

(a) Extended Horizon (T=96)

- Periodic demand pattern: $p_{t+14} = p_t$
- State space remains same, extended backward induction
- Optimal cost decreases with longer planning horizon
- **Estimated expected cost under optimal policy:** $\boxed{765.28}$

(b) 2-Day Delivery Lag

- **New state**: (x_t, u_{t-1})
- **Modified dynamics**:

$$x_{t+1} = \max(0, x_t + u_{t-1} - d_t)$$

- **Bellman equation**:

$$V(t, x, u_p) = \min_u \left[E[c(t, x, u) + V(t+1, x', u)]\right]$$

  where $x' = \max(0, x + u_p - d_t)$
- Optimal value increases by 8-12% due to delays
- **Estimated expected cost under policy:** $\boxed{148.0}$

3 Dice Game Problem

3.1 Simple game

(a) Dynamical System

- **State**: $s_t = (x_t, d_t)$
  - $x_t \in N$: Current points
  - $d_t \in \{1, 2, 3\}$: Number of dice
- **Control**: $u_t \in \{0, 1\}$
  - 0: Don't buy a new dice
  - 1: Buy a new dice (if $x_t \geq 6$ and $d_t < 3$)
- **Dynamics**:

$$s_{t+1} = \begin{cases} (x_t - 5 + M_t, d_t + 1) & \text{if } u_t = 1 \text{ ; } d_t < 3 \text{ and } x_t \geq 6 \\ (x_t + M_t, d_t) & \text{otherwise} \end{cases}$$

  where $M_t = \max(\text{rolls(t)})$

(b) Simple Heuristic Strategy

We implemented a basic heuristic strategy where the player buys a new die whenever they have at least 6 points and fewer than 3 dice. Otherwise, they do nothing. This rule is applied at each time step t, based on the current state (x, d), where:

- x: current score (points),
- d: number of dice currently held.

The heuristic strategy is defined by:

$$\pi(t, x, d) = \begin{cases} 1 & \text{if } x \geq 6 \text{ and } d < 3 \\ 0 & \text{otherwise} \end{cases}$$

This strategy reflects a greedy approach that invests in more dice as soon as it is affordable, without considering the remaining horizon or expected future rewards.

(c) Strategy Evaluation by Simulation

We implemented a Monte Carlo simulator to estimate the expected gain associated with any given strategy. The simulator runs $N = 10{,}000$ independent games, each over a 10-turn horizon, and returns:

- the average final score,
- a 95% confidence interval based on the empirical standard deviation.

The 95% confidence interval is computed using:

$$\hat{\mu} \pm 1.96 \cdot \frac{\hat{\sigma}}{\sqrt{N}}$$

For the heuristic strategy defined in part (b), we obtain:

- **Estimated expected score:** $\boxed{37.564}$
- **95% confidence interval:** $\boxed{[37.464,\ 37.664]}$

This confirms the simulator's ability to evaluate the average performance of a strategy under the game's stochastic dynamics.

(d) Maximum of Dice Distribution

For k dice, let $D_1, \ldots, D_k$) be the outcome of the k dices, and let $Z = max(D_1, \ldots, D_k)$

$$\begin{aligned} P(Z = m) &= P(Z \leq m) - P(Z \leq m-1) \\ &= P(D_1 \leq m, \ldots, D_k \leq m) - P(D_1 \leq m-1, \ldots, D_k \leq m-1) \\ &= \left(\frac{m}{6}\right)^k - \left(\frac{m-1}{6}\right)^k \end{aligned}$$

m	1 dice	2 dice	3 dice
1	$\frac{1}{6}$	$\frac{1}{36}$	$\frac{1}{216}$
2	$\frac{1}{6}$	$\frac{3}{36}$	$\frac{7}{216}$
3	$\frac{1}{6}$	$\frac{5}{36}$	$\frac{19}{216}$
4	$\frac{1}{6}$	$\frac{7}{36}$	$\frac{37}{216}$
5	$\frac{1}{6}$	$\frac{9}{36}$	$\frac{61}{216}$
6	$\frac{1}{6}$	$\frac{11}{36}$	$\frac{91}{216}$

Expected values: $E[M] = \begin{cases} 3.5 & d = 1 \\ 4.472 & d = 2 \\ 5.056 & d = 3 \end{cases}$

(e) Optimal Strategy via DP

Value function $V(t, x, d)$: Maximum expected final points from state (x, d) at turn t

$$V(T+1, x, d) = x$$

$$V(t, x, d) = \max_{u \in \{0,1\}} \sum_{m=1}^{6} P(M = m | d') V(t+1, x' + m, d')$$

where:

- $d' = \min(d + u, 3)$
- $x' = x - 5u$ (if buying)

Optimal strategy: Buy dice early when:

$$E[V(t+1, x-5+m, d+1)] > E[V(t+1, x+m, d)]$$

through the numerical optimal policy calculations is : it's optimal to buy if you can and T≤5 .

(f) Simulation Check of the Optimal Strategy

To validate the correctness of the dynamic programming result, we simulated the optimal policy π^* over $N = 10{,}000$ independent runs using the simulator defined in part (c).

The results are:

- **Dynamic programming value:** $\boxed{V_0 = 37.62}$
- **Simulated expected score:** $\boxed{37.564}$
- **95% confidence interval:** $\boxed{[37.464,\ 37.664]}$

The value obtained via simulation is very close to the dynamic programming estimate and lies entirely within the 95% confidence interval. This confirms that the optimal policy π^*, as computed by dynamic programming, performs as expected when simulated under the game's stochastic rules.

(g) Horizon Limit for Buying

It becomes suboptimal to buy a new dice when the expected gain from additional rolls no longer offsets the fixed purchase cost of 5 points.

Buying is no longer optimal when:

$$(T - t + 1)\,(E[M_{d+1}] - E[M_d]) < 5$$

This inequality compares the total expected benefit over the remaining horizon to the cost of purchasing a die.
For the first increment (from 1 to 2 dice), we have:

$$E[M_2] - E[M_1] \approx 0.972 \Rightarrow \text{critical horizon } T - t + 1 \leq \left\lfloor \frac{5}{0.972} \right\rfloor = 5$$

Therefore, buying is only optimal when there are at least 6 steps left — i.e., $t \leq 5$. So if $T \leq 8$, it's never worth buying a die.

—

(h) Unlimited Dice Case

When the number of dice is unbounded (but still only one can be bought per turn), the marginal value of buying a new die decreases as d increases.

As $d \to \infty$, we have:

$$\lim_{d\to\infty} E[M_d] = 6 \quad \text{and} \quad E[M_{d+1}] - E[M_d] \to 0$$

Consequently, the marginal gain eventually falls below the cost of 5 points. Simulations and DP results typically show that buying beyond $d = 4$ or 5 is no longer profitable.

3.2 spending dices

(a) Optimal Strategy with Max 5 Dice

+ Dynamical system

- **State space**: (t, x, d) where:
  - $t \in \{1, \ldots, 10\}$: Turn number
  - $x \in N$: Current points
  - $d \in \{1, \ldots, 5\}$: Number of dice
- **Controls**:
  - Buy decision $b \in \{0, 1\}$
  - Spend decision $s \in \{0, 1\}$
- **Bellman equation**:

$$V(t, x, d) = \max_{b \in \{0,1\}} \left[E_{m \sim \text{Dice}(d')} \max_{s \in \{0,1\}} \Big[(2^s m - 5b) + V(t+1, x + 2^s m - 5b, d' - s) \Big] \right]$$

  where $d' = \min(d + b, 5)$

+ Optimal value and optimal strategy

We solved the dynamic programming problem for a 10-turn dice game with a maximum of 5 dice allowed. At each turn, the player can either:

- Buy a new die (if they have fewer than 5 dice and at least 5 points),
- Roll all current dice, then either:
  - Add the maximum roll to their score, or
  - Spend a die to double the value of the roll (if they have at least 2 dice).

The optimal value function was computed using backward induction. The initial value at state $(t = 1, x = 0, d = 1)$ is:

$$\boxed{V(1, 0, 1) = 47.59}$$

The optimal strategy tends to:

- Buy additional dice early when possible (to increase the max roll),
- Use the "spend a die to double" option when it provides a higher expected value,
- Conserve dice toward the end of the game if the marginal gain is small.

(b) Unlimited Dice Case

When the maximum number of dice is no longer limited, the state space expands to $d \in N^+$, and the strategy becomes more aggressive with respect to buying dice — especially early in the game.

- As the number of dice increases, so does the expected value of the maximum roll:
$$E[M_d] \nearrow 6 \quad \text{as } d \to \infty$$
- However, the *marginal gain* from each new die:
$$\Delta_d = E[M_{d+1}] - E[M_d]$$
decreases rapidly with d.
- Buying becomes suboptimal when the expected gain from additional dice no longer offsets the cost (5 points). That is, the policy stops buying when:
$$(T - t + 1) \cdot \Delta_d \leq 5$$
- Thus, even with unlimited dice allowed, the *optimal policy naturally limits* the number of dice purchased — usually between 4 and 5, depending on the horizon.
- Additionally, when dice are abundant, spending one to double a roll becomes less appealing because each individual die is less valuable relative to the total pool.

(c) End-Game Selling Bonus

- The optimal value is deduced after modifying the terminal condition of the dynamic programming:
$$V(T + 1, x, d) = x + K(d)$$
- Such that the selling bonus table is as follow:

Dice (d)	1	2	3	4	5
Bonus (K)	0	2	4	5	8

- We then solved the dynamic programming problem using backward induction, incorporating the resale bonus K_d in the terminal condition. The optimal expected gain starting from state $(t = 1, x = 0, d = 1)$ is:
$$\boxed{V(1, 0, 1) = 47.71}$$

- This value improves upon the base case without resale, illustrating that adding a terminal reward based on dice ownership provides a meaningful incentive for strategic investment in dice.

  Impact on the optimal strategy:

  - Players are encouraged to preserve high-value dice near the end of the game to benefit from resale.
  - Spending a die becomes less attractive in late stages, especially if the resale value outweighs the short-term gain.
  - Early in the game, investment in dice may still be optimal to build up scoring potential.

(d) Approximating a 20-Turn Game with a 10-Turn Horizon

To approximate the 20-turn game using only a 10-turn dynamic program, we modify the terminal condition at turn 11 by adding a bonus value K_d, which represents the expected value of playing 10 more turns with d dice.

Theoretically, the continuation bonus is defined as:

$$K_d = E_x \left[V^{(20)}(11, x, d) - x \right]$$

This represents the average number of points a player can still expect to earn starting from turn 11, given that they already have x points and hold d dice.

However, in this game, the value function grows approximately linearly with the number of points x. That is:

$$V^{(20)}(11, x, d) \approx x + f(d)$$

This means that the value gained from having additional points is additive and mostly independent of the dice dynamics. As a result, the difference $V - x$ is almost constant for a given number of dice d, and we can simplify the expression by choosing $x = 0$:

$$K_d \approx V^{(20)}(11, 0, d)$$

This approximation allows us to compute K_d directly by solving the 20-turn dynamic program once and reading off the value at turn 11 with 0 points.

Then, in the 10-turn model, we set the terminal condition as:

$$V^{(10)}(11, x, d) = x + K_d$$

This adjustment allows the 10-turn model to mimic the long-term rewards of a full 20-turn game without having to solve the entire problem repeatedly.

Estimated resale bonuses based on 20-turn value function:

Number of Dice d	**Estimated Bonus** $K(d)$
1	47.59
2	56.98
3	64.13
4	70.58
5	76.39

(e) Strategy for a 10^{10}-Turn Game

Solving a dynamic program over 10^{10} steps is computationally infeasible. However, we can approximate a near-optimal strategy using the concept of **stationary behavior** in long-horizon problems.

We bear in mind that in very long games, players care more about long-term rewards than immediate ones, and so the effect of the initial state or early randomness fades with time as the game progresses. Hence, for large horizons, if the value function and the optimal policy tend to stabilize, the decisions made in later steps become nearly time-invariant. This allows us to extract a good approximation of the infinite-horizon behavior from a short-horizon computation.

Practical approach:

1. Solve a smaller dynamic program (e.g., 20 steps), using a terminal condition such as:
$$V(21, x, d) = x + K_d$$
where K_d reflects the expected future reward of having d dice.

2. Examine the policy in the last few steps (e.g., turns 15–20). If the optimal actions become stable (i.e., same decisions for similar states), we can assume the strategy has converged.

3. Use this **stationary policy** repeatedly as a strategy for any large horizon, including 10^{10} turns.