Stochastic Dynamic Programming

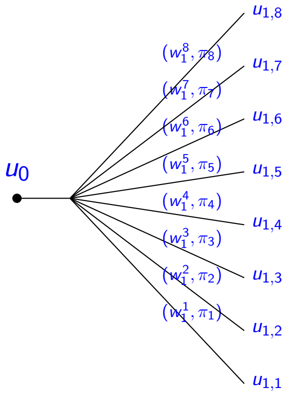
V. Leclère (ENPC)
F. Pacaud, T.Rigaut (Efficacity)

March 14, 2017

Contents

Contents

Where do we come from: two-stage programming



- We take decisions in two stages

$$u_0 \rightsquigarrow \mathbf{W}_1 \rightsquigarrow \mathbf{U}_1 \ ,$$

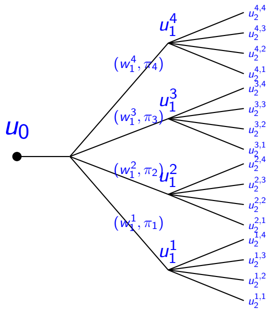
with $\mathbf{U}_1$: recourse decision .

- On a tree, it resumes to solve the extensive formulation:

$$\min_{u_0, u_{1,s}} \sum_{s \in \mathbb{S}} \pi_s \Big[\big\langle c_s \ , u_0 \big\rangle + \big\langle p_s \ , u_{1,s} \big\rangle \Big] \ .$$

We have as many $u_{1,s}$ as scenarios!

Extending two-stage to multistage programming


u_0
(w_1^4, π_4)
(w_1^3, π_3)
(w_1^2, π_2)
(w_1^1, π_1)
u_1^4
u_1^3
u_1^2
u_1^1
$u_2^{4,4}$ $u_2^{4,3}$ $u_2^{4,2}$ $u_2^{4,1}$
$u_2^{3,4}$ $u_2^{3,3}$ $u_2^{3,2}$ $u_2^{3,1}$
$u_2^{2,4}$ $u_2^{2,3}$ $u_2^{2,2}$ $u_2^{2,1}$
$u_2^{1,4}$ $u_2^{1,3}$ $u_2^{1,2}$ $u_2^{1,1}$

$$\min_{\mathbf{U}} \ \mathbb{E}\big(j(\mathbf{U},\mathbf{W})\big)$$

$$\mathbf{U} = (\mathbf{U}_0, \cdots, \mathbf{U}_T)$$

$$\mathbf{W} = (\mathbf{W}_1, \cdots, \mathbf{W}_T)$$

We take decisions in T stages

$$\mathbf{W}_0 \rightsquigarrow \mathbf{U}_0 \rightsquigarrow \mathbf{W}_1 \rightsquigarrow \mathbf{U}_1 \rightsquigarrow \cdots \rightsquigarrow \mathbf{W}_T \rightsquigarrow \mathbf{U}_T \ .$$

Introducing the non-anticipativity constraint

We do not know what holds behind the door.

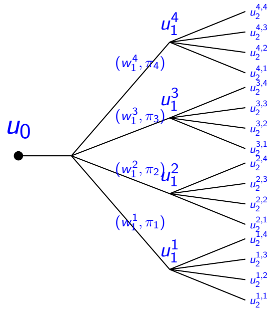
Non-anticipativity

At time t, decisions are taken sequentially, only knowing the past realizations of the perturbations.

Mathematically, this is equivalent to say that at time t, the decision $\mathbf{U}_t$ is

1. a function of past noises
$$\mathbf{U}_t = \pi_t(\mathbf{W}_0, \cdots, \mathbf{W}_t) \ ,$$
2. taken knowing the available information,
$$\sigma(\mathbf{U}_t) \subset \sigma(\mathbf{W}_0, \cdots, \mathbf{W}_t) \ .$$

Multistage extensive formulation approach


u_0
(w_1^1, π_1)
(w_1^2, π_2)
(w_1^3, π_3)
(w_1^4, π_4)
u_1^1
u_1^2
u_1^3
u_1^4
$u_2^{4,4}$ $u_2^{4,3}$ $u_2^{4,2}$ $u_2^{4,1}$
$u_2^{3,4}$ $u_2^{3,3}$ $u_2^{3,2}$ $u_2^{3,1}$
$u_2^{2,4}$ $u_2^{2,3}$ $u_2^{2,2}$ $u_2^{2,1}$
$u_2^{1,4}$ $u_2^{1,3}$ $u_2^{1,2}$ $u_2^{1,1}$

Assume that $w_t \in \mathbb{R}^{n_w}$ can take n_w values and that $U_t(x)$ can take n_u values.

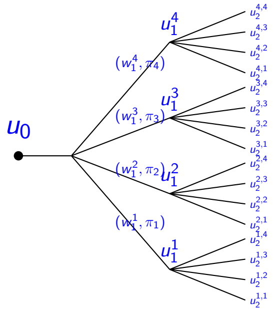
Then, considering the extensive formulation approach, we have

- n_w^T scenarios.
- $(n_w^{T+1} - 1)/(n_w - 1)$ nodes in the tree.
- Number of variables in the optimization problem is roughly $n_u \times (n_w^{T+1} - 1)/(n_w - 1) \approx n_u n_w^T$.

The complexity grows exponentially with the number of stage. :-(

A way to overcome this issue is to compress information!

Multistage extensive formulation approach



Assume that $w_t \in \mathbb{R}^{n_w}$ can take n_w values and that $U_t(x)$ can take n_u values.

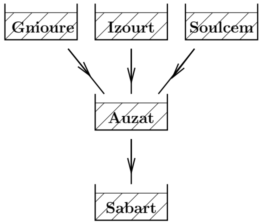
Then, considering the extensive formulation approach, we have

- n_w^T scenarios.
- $(n_w^{T+1} - 1)/(n_w - 1)$ nodes in the tree.
- Number of variables in the optimization problem is roughly $n_u \times (n_w^{T+1} - 1)/(n_w - 1) \approx n_u n_w^T$.

The complexity grows exponentially with the number of stage. :-(

A way to overcome this issue is to compress information!

Illustrating extensive formulation with the damsvalley example



- 5 interconnected dams
- 5 controls per timesteps
- 52 timesteps (one per week, over one year)
- $n_w = 10$ noises for each timestep

We obtain 10^{52} scenarios, and $\approx 5.10^{52}$ constraints in the extensive formulation ... Estimated storage capacity of the Internet: 10^{24} bytes.

Contents

Compressing information inside a state

Due to non-anticipativity constraint, decisions are function of previous history:

$$\sigma(\mathbf{U}_t) = \pi_t(\mathbf{W}_0, \cdots, \mathbf{W}_t) .$$

As the number of timesteps increases, the computation of the policy π_t becomes more and more complicated.

A solution is to compute decisions as function of a sufficient aggregated information called state (and denoted by $\mathbf{X}_t$):

$$\sigma(\mathbf{U}_t) = \psi_t(\mathbf{X}_t) .$$

This is equivalent to find a sufficient statistic for the process $(\mathbf{W}_0, \cdots, \mathbf{W}_t)$.

Contents

Stochastic Controlled Dynamic System

A discrete time controlled stochastic dynamic system is defined by its *dynamic*

$$\mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1})$$

and initial state

$$\mathbf{X}_0 = \mathbf{W}_0$$

The variables

- $\mathbf{X}_t$ is the *state* of the system,
- $\mathbf{U}_t$ is the *control* applied to the system at time t,
- $\mathbf{W}_t$ is an exogeneous noise.

Usually, $\mathbf{X}_t \in \mathbb{X}_t$ and $\mathbf{U}_t$ beglongs to a set depending upon the state: $\mathbf{U}_t \in \mathbb{U}_t(\mathbf{X}_t)$.

Examples

- Stock of water in a dam:
  - $\mathbf{X}_t$ is the amount of water in the dam at time t,
  - $\mathbf{U}_t$ is the amount of water turbined at time t,
  - $\mathbf{W}_{t+1}$ is the inflow of water in $[t, t+1[$.
- Boat in the ocean:
  - $\mathbf{X}_t$ is the position of the boat at time t,
  - $\mathbf{U}_t$ is the direction and speed chosen for $[t, t+1[$,
  - $\mathbf{W}_{t+1}$ is the wind and current for $[t, t+1[$.
- Subway network:
  - $\mathbf{X}_t$ is the position and speed of each train at time t,
  - $\mathbf{U}_t$ is the acceleration chosen at time t,
  - $\mathbf{W}_{t+1}$ is the delay due to passengers and incident on the network for $[t, t+1[$.

Optimization Problem

We want to solve the following optimization problem

$$\begin{aligned} \min \quad & \mathbb{E}\Big(\sum_{t=0}^{T-1} L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}) + K(\mathbf{X}_T)\Big) \\ s.t. \quad & \mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}), \qquad \mathbf{X}_0 = \mathbf{W}_0 \\ & \mathbf{U}_t \in \mathbb{U}_t(\mathbf{X}_t) \\ & \sigma(\mathbf{U}_t) \subset \sigma\big(\mathbf{W}_0, \cdots, \mathbf{W}_t\big) \end{aligned}$$

1. We want to minimize the expectation of the sum of costs.
2. The system follows a dynamic given by the function f_t.
3. There are constraints on the controls.
4. The controls are functions of the past noises (= non-anticipativity).

Optimization Problem

We want to solve the following optimization problem

$$\begin{aligned}
\min \quad & \mathbb{E}\Big(\sum_{t=0}^{T-1} L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}) + K(\mathbf{X}_T) \Big) \\
s.t. \quad & \mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}), \qquad \mathbf{X}_0 = \mathbf{W}_0 \\
& \mathbf{U}_t \in \mathbb{U}_t(\mathbf{X}_t) \\
& \sigma(\mathbf{U}_t) \subset \sigma\big(\mathbf{W}_0, \cdots, \mathbf{W}_t\big)
\end{aligned}$$

1. We want to minimize the expectation of the sum of costs.
2. The system follows a dynamic given by the function f_t.
3. There are constraints on the controls.
4. The controls are functions of the past noises
(= non-anticipativity).

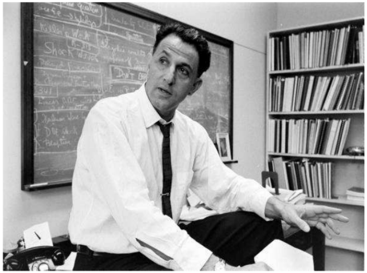
Optimization Problem with independence of noises

If noises at time independent, the optimization problem is equivalent to

$$\begin{aligned}
\min \quad & \mathbb{E}\Big(\sum_{t=0}^{T-1} L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}) + K(\mathbf{X}_T)\Big) \\
s.t. \quad & \mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}), \qquad \mathbf{X}_0 = \mathbf{W}_0 \\
& \mathbf{U}_t \in \mathbb{U}_t(\mathbf{X}_t) \\
& \mathbf{U}_t = \psi_t(\mathbf{X}_t)
\end{aligned}$$

Contents

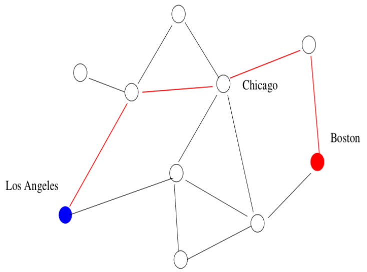
Bellman's Principle of Optimality



Richard Ernest Bellman (August 26, 1920 – March 19, 1984)

An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision (Richard Bellman)

The shortest path on a graph illustrates Bellman's Principle of Optimality



For an auto travel analogy, suppose that the fastest route from Los Angeles to Boston passes through **Chicago**.
The principle of optimality translates to obvious fact that the Chicago to Boston portion of the route is also the fastest route for a trip that starts from Chicago and ends in Boston.
(Dimitri P. Bertsekas)

Idea behind dynamic programming

If noises are time independent, then

1. The cost to go at time t depends only upon the current state.
2. We can compute recursively the cost to go for each position, starting from the terminal state and computing optimal trajectories backward.

Optimal cost-to-go of being in state x at time t is:

$$V_t(x) = \min_{u \in \mathbb{U}_t(x)} \mathbb{E}\Big(\underbrace{L_t(x,u,\mathbf{W}_{t+1})}_{\text{instantaneous cost}} + \underbrace{V_{t+1} \circ f_t(x,u,\mathbf{W}_{t+1})}_{\text{cost to be in } \mathbf{X}_{t+1} \text{ at time t+1}} \Big) .$$

At time t, V_{t+1} gives the cost of the future. Dynamic Programming is a time decomposition method.

Idea behind dynamic programming

If noises are time independent, then

1. The cost to go at time t depends only upon the current state.
2. We can compute recursively the cost to go for each position, starting from the terminal state and computing optimal trajectories backward.

Optimal cost-to-go of being in state x at time t is:

$$V_t(x) = \min_{u \in \mathbb{U}_t(x)} \mathbb{E}\Big(\underbrace{L_t(x, u, \mathbf{W}_{t+1})}_{\text{instantaneous cost}} + \underbrace{V_{t+1} \circ f_t\big(x, u, \mathbf{W}_{t+1}\big)}_{\text{cost to be in } \mathbf{X}_{t+1} \text{ at time t+1}} \Big) .$$

At time t, V_{t+1} gives the cost of the future. Dynamic Programming is a time decomposition method.

Dynamic Programming Principle

Assume that the noises $\mathbf{W}_t$ are time-independent and exogeneous. The Bellman's equation writes

$$\begin{cases} V_T(x) &= K(x) \\ V_t(x) &= \min_{u \in \mathbb{U}_t(x)} \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1}) + V_{t+1} \circ \underbrace{f_t\big(x, u, \mathbf{W}_{t+1}\big)}_{\text{"}\mathbf{X}_{t+1}\text{"}} \Big) \end{cases}$$

Decisions are taken as $\mathbf{U}_t = \pi_t(\mathbf{X}_t)$, with

$$\pi_t(x) \in \arg\min_{u \in \mathbb{U}_t(x)} \mathbb{E}\Big(\underbrace{L_t(x, u, \mathbf{W}_{t+1})}_{\text{current cost}} + \underbrace{V_{t+1} \circ f_t\big(x, u, \mathbf{W}_{t+1}\big)}_{\text{future costs}} \Big) ,$$

Dynamic Programming Principle

Assume that the noises $\mathbf{W}_t$ are time-independent and exogeneous. The Bellman's equation writes

$$\begin{cases} V_T(x) &= K(x) \\ V_t(x) &= \min_{u \in \mathbb{U}_t(x)} \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1}) + V_{t+1} \circ \underbrace{f_t\big(x, u, \mathbf{W}_{t+1}\big)}_{\text{"}\mathbf{X}_{t+1}\text{"}} \Big) \end{cases}$$

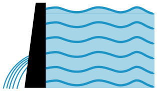
Decisions are taken as $\mathbf{U}_t = \pi_t(\mathbf{X}_t)$, with

$$\pi_t(x) \in \arg\min_{u \in \mathbb{U}_t(x)} \mathbb{E}\Big(\underbrace{L_t(x, u, \mathbf{W}_{t+1})}_{\text{current cost}} + \underbrace{V_{t+1} \circ f_t\big(x, u, \mathbf{W}_{t+1}\big)}_{\text{future costs}} \Big) ,$$

Interpretation of Bellman Value Function

The Bellman's value function $V_{t_0}(x)$ can be interpreted as the value of the problem starting at time t_0 from the state x. More precisely we have

$$
\begin{aligned}
V_{t_0}(x) = \min \quad & \mathbb{E}\Big(\sum_{t=t_0}^{T-1} L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}) + K(\mathbf{X}_T) \Big) \\
s.t. \quad & \mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}), \qquad \mathbf{X}_{t_0} = x \\
& \mathbf{U}_t \in \mathbb{U}_t(\mathbf{X}_t) \\
& \sigma(\mathbf{U}_t) \subset \sigma\big(\mathbf{W}_0, \cdots, \mathbf{W}_t\big)
\end{aligned}
$$

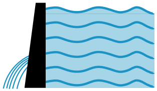


Ex: Economists can view V as a way to evaluate a stock (= value of water for a dam)

Interpretation of Bellman Value Function

The Bellman's value function $V_{t_0}(x)$ can be interpreted as the value of the problem starting at time t_0 from the state x. More precisely we have

$$
\begin{aligned}
V_{t_0}(x) = \min \quad & \mathbb{E}\Big(\sum_{t=t_0}^{T-1} L_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}) + K(\mathbf{X}_T) \Big) \\
s.t. \quad & \mathbf{X}_{t+1} = f_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1}), \qquad \mathbf{X}_{t_0} = x \\
& \mathbf{U}_t \in \mathbb{U}_t(\mathbf{X}_t) \\
& \sigma(\mathbf{U}_t) \subset \sigma(\mathbf{W}_0, \cdots, \mathbf{W}_t)
\end{aligned}
$$



Ex: Economists can view V as a way to evaluate a stock (= value of water for a dam)

Contents

Dynamic Programming Algorithm - Discrete Case

```
**Data:** Problem parameters
**Result:** optimal control and value;
$V_T \equiv K$ ;
for $t : T-1 \to 0$ do
    for $x \in \mathbb{X}_t$ do
        $V_t(x) = \min_{u \in U_t(x)} \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1}) + V_t(f_t(x, u, \mathbf{W}_{t+1}))\Big)$
```

Algorithm 1: We iterate over the discretized state space

Dynamic Programming Algorithm - Discrete Case

```
Data: Problem parameters
Result: optimal control and value;
$V_T \equiv K$ ;
for $t : T-1 \to 0$ do
    for $x \in \mathbb{X}_t$ do
        $V_t(x) = \infty$;
        for $u \in U_t(x)$ do
            $v_u = \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1}) + V_t(f_t(x, u, \mathbf{W}_{t+1}))\Big)$ if
            $v_u < V_t(x)$ then
                $V_t(x) = v_u$ ;
                $\pi_t(x) = u$ ;
```

Algorithm 2: We iterate over the discretized control space

Dynamic Programming Algorithm - Discrete Case

```
Data: Problem parameters
Result: optimal control and value;
$V_T \equiv K$ ;
for $t : T-1 \to 0$ do
    for $x \in \mathbb{X}_t$ do
        $V_t(x) = \infty$;
        for $u \in U_t(x)$ do
            $v_u = 0$;
            for $w \in \mathbb{W}_t$ do
                $v_u = v_u + \mathbb{P}\{w\}\big(L_t(x,u,w) + V_{t+1}(f_t(x,u,w))\big)$;
            if $v_u < V_t(x)$ then
                $V_t(x) = v_u$ ;
                $\pi_t(x) = u$ ;
```

Algorithm 3: Classical stochastic dynamic programming algorithm

3 curses of dimensionality

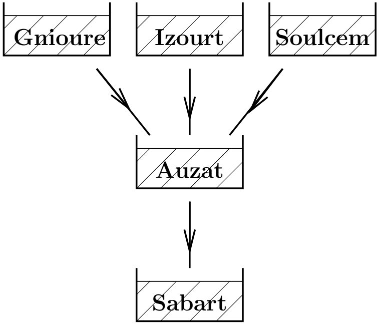
Complexity $= O(T \times |\mathbb{X}_t| \times |\mathbb{U}_t| \times |\mathbb{W}_t|)$

This is linear in the number of time steps :-)

But we have 3 curses of dimensionality :-(:

1. State. Complexity is exponential in the dimension of $\mathbb{X}_t$
2. Decision. Complexity is exponential in the dimension of $\mathbb{U}_t$
3. Expectation. Complexity is exponential in the dimension of $\mathbb{W}_t$

Illustrating dynamic programming with the damsvalley example



Illustrating the curse of dimensionality

We are in dimension 5 (not so high in the world of big data!) with 52 timesteps (common in energy management) plus 5 controls and 5 independent noises.

1. We discretize each state's dimension in 100 values: $|\mathbb{X}_t| = 100^5 = 10^{10}$
2. We discretize each control's dimension in 100 values: $|\mathbb{U}_t| = 100^5 = 10^{10}$
3. We use optimal quantization to discretize the noises' space in 10 values: $|\mathbb{W}_t| = 10$

Number of flops: $\mathcal{O}(52 \times 10^{10} \times 10^{10} \times 10) \approx \mathcal{O}(10^{23})$.
In the TOP500, the best computer computes 10^{17} flops/s.
Even with the most powerful computer, it takes at least 12 days to solve this problem.

Contents

DP on a Markov Chain

- Sometimes it is easier to represent a problem as a controlled Markov Chain
- Dynamic Programming equation can be computed directly, without expliciting the control.
- Let's work out an example...

Controlled Markov Chain

- A controlled Markov Chain is controlled stochastic dynamic system with independent noise $(\mathbf{W}_t)_{t\in\mathbb{Z}}$, where the dynamic and the noise are left unexplicited.
- What is given is the *transition probability*

$$\pi_t^u(x,y) := \mathbb{P}\Big(\mathbf{X}_{t+1} = y \mid \mathbf{X}_t = x, \mathbf{U}_t = u\Big).$$

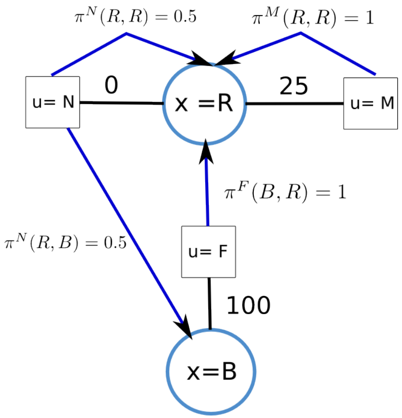
- In this case the cost are given as a function of the current stage, the next stage and the control.
- The Dynamic Programming Equation then reads (assume finite state)

$$V_t(x) = \min_u \sum_{y\in\mathbb{X}_{t+1}} \pi_t^u(x,y)\Big[L_t^u(x,y) + V_{t+1}(y)\Big].$$

Example

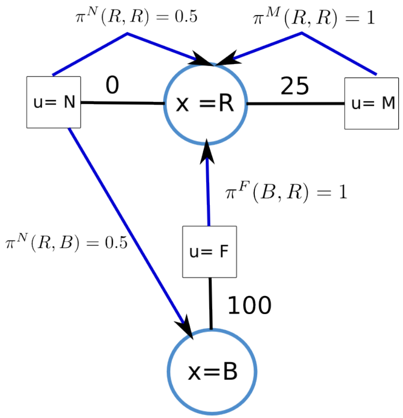
Consider a machine that has two states : running (R) and broken (B). If it is broken we need to fix it (F) for a cost of 100. If it is running there are two choices: maintaining it (M), or not maintaining (N). If we maintain, the cost is 25 and the machine stay running with probability $\pi^M(R,R) = 1$; if we do not maintain there is a probability of $\pi^N(R,B) = 0.5$ of breaking it (or keep it running). We need to have it running for 3 periods.

Controlled Markov Chain



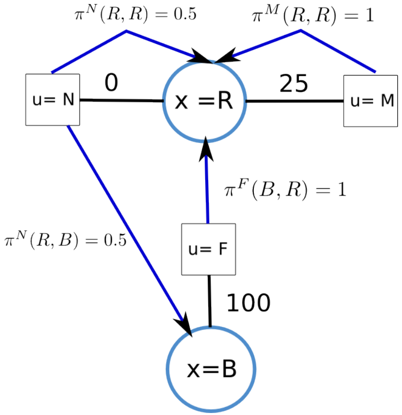
	V_0	V_1	V_2	V_3	V_4
R					0
B					0

Controlled Markov Chain



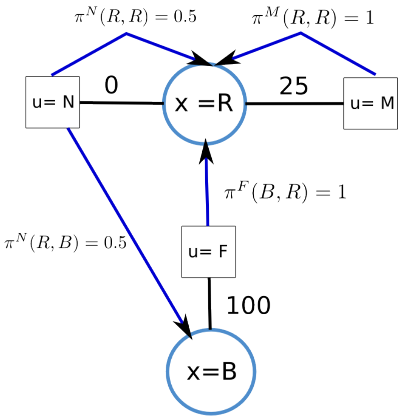
	V_0	V_1	V_2	V_3	V_4
R					0
B					0

Controlled Markov Chain



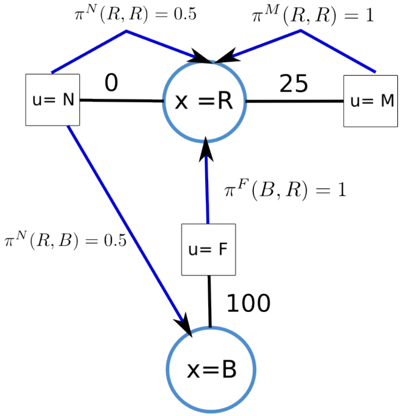
	V_0	V_1	V_2	V_3	V_4
R				$\min\left\{25+0, 0+(0+0)/2\right\}$	0
B				$100+0$	0

Controlled Markov Chain



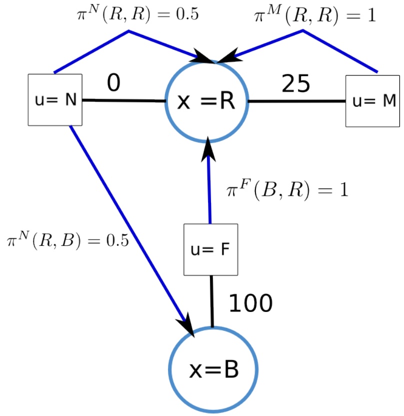
	V_0	V_1	V_2	V_3	V_4
R			$\min\left\{25+0, 0+(0+100)/2\right\}$	0	0
B			$100+0$	100	0

Controlled Markov Chain



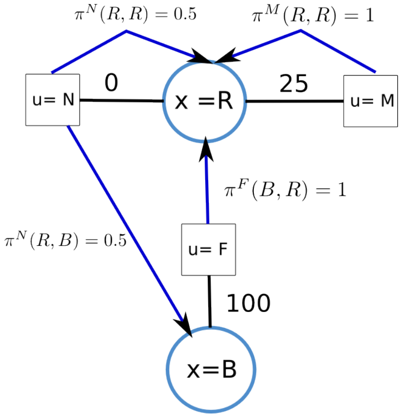
	V_0	V_1	V_2	V_3	V_4
R		$\min\left\{25+25, 0+(25+100)/2\right\}$	25	0	0
B		$100+25$	100	100	0

Controlled Markov Chain



	V_0	V_1	V_2	V_3	V_4
R	$\min\{25+50, 0+(50+125)/2\}$	50	25	0	0
B	$100+50$	125	100	100	0

Controlled Markov Chain



	V_0	V_1	V_2	V_3	V_4
R	75	50	25	0	0
B	150	125	100	100	0

Contents

Computing a decision online

Algorithm: Offline value functions precomputation + Online open loop reoptimization

Offline: We produce value functions with Bellman equation:

$$V_t(x) = \min_{u \in \mathbb{U}_t} \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1}) + V_{t+1}(f_t(x, u, \mathbf{W}_{t+1})) \Big)$$

Online: At time t, knowing x_t we plug the computed value function V_{t+1} as future cost

$$u_t \in \arg\min_{u \in \mathbb{U}_t} \mathbb{E}\Big(L_t(x_t, u, \mathbf{W}_{t+1}) + V_{t+1}(f_t(x_t, u, \mathbf{W}_{t+1})) \Big)$$

Independence of noises

- The Dynamic Programming equation requires only the time-independence of noises.
- This can be relaxed if we consider an extended state.
- Consider a dynamic system driven by an equation

$$\mathbf{Y}_{t+1} = f_t(\mathbf{Y}_t, \mathbf{U}_t, \varepsilon_{t+1})$$

where the random noise ε_t is an AR-1 process :

$$\varepsilon_t = \alpha_t \varepsilon_{t-1} + \beta_t + \mathbf{W}_t,$$

$\{\mathbf{W}_t\}_{t\in\mathbb{Z}}$ being independent.

- Then $\mathbf{Y}_t$ is called the physical state of the system and DP can be used with the information state $\mathbf{X}_t = (\mathbf{Y}_t, \varepsilon_t)$.
- Generically speaking, if the noise $\mathbf{W}_t$ is exogeneous (not affected by decisions $\mathbf{U}_t$), then we can always apply Dynamic Programming with the state $(\mathbf{X}_t, \mathbf{W}_1, \dots, \mathbf{W}_t)$.

State augmentation limits

Because of the curse of dimensionality it might be impossible to take into account correlation by augmenting the state variable.

Practitioners often ignore noise dependence for the value functions computation but use dependence information during online reoptimization.

We present this technique in a following industrial case presentation

Conclusion

- Multistage stochastic programming fails to handle large number of timesteps.
- Dynamic Programming overcomes this difficulty while compressing information inside a state $\mathbf{X}$.
- Dynamic Programming computes backward a set of value functions $\{V_t\}$, corresponding to the optimal cost of being at a given position at time t.
- Numerically, DP is limited by the curse of dimensionality and its performance are deeply related to the discretization of the look-up table used.
- Other methods exist to compute the value functions without look-up table (Approximate Dynamic Programming, SDDP).

Independence of noises: AR-1 case

- Consider a dynamic system driven by an equation $\mathbf{Y}_{t+1} = f_t(\mathbf{Y}_t, \mathbf{U}_t, \boldsymbol{\varepsilon}_{t+1})$ where the random noise $\boldsymbol{\varepsilon}_t$ is an AR-1 process : $\boldsymbol{\varepsilon}_t = \alpha_t \boldsymbol{\varepsilon}_{t-1} + \beta_t + \mathbf{W}_{t+1}$, $\{\mathbf{W}_t\}_{t\in\mathbb{Z}}$ being independent.
- Define the information state $\mathbf{X}_t = (\mathbf{Y}_t, \boldsymbol{\varepsilon}_t)$.
- Then we have

$$\mathbf{X}_{t+1} = \begin{pmatrix} f_t(\mathbf{Y}_t, \mathbf{U}_t, \alpha_t \boldsymbol{\varepsilon}_t + \beta_t + \mathbf{W}_{t+1}) \\ \alpha_t \boldsymbol{\varepsilon}_t + \beta_t + \mathbf{W}_{t+1} \end{pmatrix} = \tilde{f}_t(\mathbf{X}_t, \mathbf{U}_t, \mathbf{W}_{t+1})$$

- And we have the following DP equation

$$V_t\binom{y}{\varepsilon} = \min_{u \in U_t(x)} \mathbb{E}\Big(L_t(y, u, \underbrace{\alpha_t \varepsilon + \beta_t + \mathbf{W}_{t+1}}_{"\boldsymbol{\varepsilon}_{t+1}"}) + V_{t+1} \circ \underbrace{\tilde{f}_t\big(x, u, \mathbf{W}_{t+1}\big)}_{"\mathbf{X}_{t+1}"}$$

Dynamic Programming : Discretization-Interpolation

- The DP equation holds :

$$V_t(x) = \min_{u \in U_t(x)} \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1} + V_{t+1} \circ f_t(x, u, \mathbf{W}_{t+1})\Big).$$

- But computation is impractical in a continuous space. Simplest solution : discretization and interpolation.
- We choose a finite set $\mathbb{X}_t^D \subset \mathbb{X}_t$ where we will compute (an approximation of) the Bellman value V_t.
- We approximate the Bellman value at time t by interpolating these value.

Dynamic Programming : Discretization-Interpolation

```
Data: Problem parameters, discretization,
          one-stage solver, interpolation operator;
Result: approximation of optimal value;
$\tilde{V}_T \equiv K$ ;
for $t : T-1 \to 0$ do
    for $x \in \mathbb{X}_t^D$ do
        $\tilde{V}_t(x) := \min_{u \in U_t(x)} \mathbb{E}\Big(L_t(x,u,\mathbf{W}_{t+1}) + \tilde{V}_{t+1} \circ f_t(x,u,\mathbf{W}_{t+1})\Big)$;
    Define $\tilde{V}_t$ by interpolating $\{\tilde{V}_t(x) \mid x \in \mathbb{X}_t^D\}$;
```

Algorithm 4: Dynamic Programming Algorithm (Continuous)

The strategy obtained is given by

$$\pi_t(x) \in \arg\min_{u \in U_t(x)} \mathbb{E}\Big(L_t(x,u,\mathbf{W}_{t+1}) + \tilde{V}_{t+1} \circ f_t(x,u,\mathbf{W}_{t+1})\Big).$$

Numerical considerations

- The discrete case algorithm a look-up table of optimal control for every possible state *offline*.
- In the continuous case we focus on computing *offline* an approximation of the value function V_t and derive the optimal control *online* by solving a one-step problem, solved only at the current state :

$$\pi_t(x) \in \underset{u \in U_t(x)}{\arg\min}\, \mathbb{E}\Big(L_t(x, u, \mathbf{W}_{t+1}) + V_{t+1} \circ f_t\big(x, u, \mathbf{W}_{t+1}\big) \Big)$$

- The field of Approximate DP gives methods for computing those approximate value function (decomposed on a base of functions).
- The simpler one consisting in discretizing the state, and then interpolating the value function.

Independence of noises

- The Dynamic Programming equation requires only the time-independence of noises.
- This can be relaxed if we consider an extended state.
- Consider a dynamic system driven by an equation

$$\mathbf{Y}_{t+1} = f_t(\mathbf{Y}_t, \mathbf{U}_t, \varepsilon_{t+1})$$

where the random noise ε_t is an AR-1 process :

$$\varepsilon_t = \alpha_t \varepsilon_{t-1} + \beta_t + \mathbf{W}_t,$$

$\{\mathbf{W}_t\}_{t\in\mathbb{Z}}$ being independent.

- Then $\mathbf{Y}_t$ is called the physical state of the system and DP can be used with the information state $\mathbf{X}_t = (\mathbf{Y}_t, \varepsilon_t)$.
- Generically speaking, if the noise $\mathbf{W}_t$ is exogeneous (not affected by decisions $\mathbf{U}_t$), then we can always apply Dynamic Programming with the state $(\mathbf{X}_t, \mathbf{W}_1, \dots, \mathbf{W}_t)$.