Dam Optimal Management under Uncertainty

P. Carpentier, J.-P. Chancelier, M. De Lara and V. Leclère

April 11, 2018

Contents

Abstract

In this practical work, you will play the role of a dam manager. You will try to maximize the mean intertemporal payoff obtained by selling hydropower produced by water releases, when the water inflows (rain, snowmelt, outflows from upper dams) are supposed to be random and the energy prices deterministic. You will propose different water turbinated policies and you will compute their expected payoff. Then, you will compute the optimal payoff and display an optimal policy. At last, you will consider different variations around this problem: highlighting the information structure influence by comparing the "decision-hazard" and the "hazard-decision" settings, evaluating a "fair final value of the water" or testing robustness with respect to inflow probability distribution.

1 Problem statement and data

We consider a dam manager intending to maximize the intertemporal payoff obtained by selling power produced by water releases, when the water inflows (rain, snowmelt, outflows from upper dams) are random and the energy prices are supposed to be deterministic.

1.1 Dam dynamics

We model the dynamics of the water volume in a dam by a function f defined as follows[1]

$$\underbrace{x(t+1)}_{\text{future volume}} = \min\left\{\overline{x},\ \underbrace{x(t)}_{\text{volume}} - \underbrace{u(t)}_{\text{turbined}} + \underbrace{w(t+1)}_{\text{inflow}}\right\},$$

$$= f\big(t, x(t), u(t), w(t+1)\big) \tag{1}$$

where

- time $t \in \mathbb{T} = \{t_i, t_i+1, \ldots, t_f\}$ is discrete (months in this case), and t denotes the beginning of the period $[t, t+1[$; we call t_i the *initial time* and t_f the *horizon*;
- $x(t)$ *volume of water* in the dam at the beginning of period $[t, t+1[$, belonging to a *finite* set $\mathbb{X} \subset [\,\underline{x}\,,\,\overline{x}\,]$, where $\underline{x}$ (resp. $\overline{x}$) is the minimum (resp. maximum) dam volume;
- $u(t)$ *turbinated outflow volume* during $[t, t+1[$, decided at the beginning of the time period $[t, t+1[$, and belonging to a *finite* set $\mathbb{U} \subset [\,\underline{u}\,,\,\overline{u}\,]$, where $\underline{u}$ (resp. $\overline{u}$) is the minimum (resp. maximum) volume which can be turbinated by time unit; we assume that the minimum value $\underline{u}$ is equal to zero:
$$0 \leq u(t) \leq \overline{u}\,. \tag{2}$$
- $w(t)$ *inflow water volume* (rain, snowmelt, etc.) during the time period $[t-1, t[$, belonging to the *finite* set $\mathbb{W}(t) \subset [\underline{w}(t)\,,\,\overline{w}(t)]$, where $\underline{w}(t)$ (resp. $\overline{w}(t)$) is the minimum (resp. maximum) possible inflow at time t; we assume that $\underline{w}(t) \geq 0$ for all t; the dependence of the sets $\mathbb{W}(t)$ upon time t allows to take into account seasonal effects (e.g. more rain in autumn).

The dam manager is supposed to make a decision at each $t \in \mathbb{T}$, here turbinating $u(t)$ at the beginning of the period $[t, t+1[$, *before knowing* the water inflow $w(t+1)$: such a setup is called *decision-hazard* setting. We add the following constraint on the turbinated water:

$$u(t) \leq x(t)\,. \tag{3}$$

Constraint (3) ensures $u(t)$ is feasible, that is, such that the dam volume remains greater or equal to its minimum value $\underline{x}$ (remember that any inflow $w(t)$ is nonnegative).

[1] The min operator in the definition of the dynamics ensures that the dam volume always remains less than or equal to its maximum value $\overline{x}$.

1.2 Criterion: intertemporal payoff

The manager problem is one of *payoff maximization*. At each time t, the turbinated water $u(t)$ produces electricity which is sold on markets at a price $p(t)$. The associated financial income is modeled by a linear function $L(t,u) = p(t)u$, leading to the *instantaneous payoff*

$$L\big(t,u(t)\big) = p(t)u(t)\ . \tag{4}$$

Moreover, the volume $x(t_f)$ remaining in the dam at the horizon t_f is valued (this is the so-called *final value of water*) using a quadratic function K, leading to the *final payoff*

$$K\big(x(t_f)\big) = -\alpha\Big(\min\big\{0, x(t_f) - x_{\rm ref}\big\}\Big)^2\ . \tag{5}$$

Here, α is a coefficient and $x_{\rm ref}$ is a given reference volume for the dam at horizon t_f. The final payoff is negative if and only if $x(t_f) < x_{\rm ref}$. From t_i to t_f, the payoffs sum up to

$$\sum_{t=t_i}^{t_f-1} L\big(t,u(t)\big) + K\big(x(t_f)\big)\ . \tag{6}$$

The sequence of prices $\big(p(t_i),\dots,p(t_f-1)\big)$ is supposed to be deterministic, hence known in advance at initial time t_i where the optimization problem is posed.

1.3 Probabilistic model for water inflows

A *water inflow scenario* is a sequence of inflows in the dam from time t_i up to $t_f - 1$

$$w(\cdot) := \big(w(t_i),\dots,w(t_f-1)\big)\ . \tag{7}$$

We model water inflow scenarios using a sequence of random variables with a known joint probability distribution $\mathbb{P}$ such that

- the random variables $\big(w(t_i),\dots,w(t_f-1)\big)$ are independent,
- each random variable $w(t)$ follows a uniform probability distribution[2] on the finite set $\mathbb{W}(t)$:

$$\mathbb{P}_{w(t)}\{w\} = \frac{1}{\mathrm{card}\big(\mathbb{W}(\mathrm{t})\big)}\ , \quad \forall w \in \mathbb{W}(t)\ . \tag{8}$$

Notice that the random variables $\big(w(t_i),\dots,w(t_f-1)\big)$ are independent, but that they are not identically distributed. Independence is a key assumption to obtain the dynamic programming equation (18) with state x.

[2]The assumption of uniform law is made for the sake of simplicity and can be relaxed without difficulty.

1.4 Numerical data

Time. We consider a montly management (the interval $[t, t+1[$ represents one month) over one year, so that

$$t_i = 1 \qquad \text{and} \qquad t_f = 13\ . \tag{9}$$

State and control. Concerning the dam volume and the turbinated water, we consider the following bounds:

$$\underline{x} = 0\ \text{hm}^3\ , \quad \overline{x} = 80\ \text{hm}^3\ , \quad \underline{u} = 0\ \text{hm}^3\ , \quad \overline{u} = 40\ \text{hm}^3\ . \tag{10}$$

We assume that the dam volume is discretized using a stepsize $\delta x = 2$ hm^3, whereas the stepsize used for discretizing the control is $\delta u = 8$ hm^3 (multiple of δx). Accordingly, the sets containing the possible values of the state and the control are

$$\mathbb{X} = \big\{0, 2, 4, \ldots, 80\big\} \qquad \text{and} \qquad \mathbb{U} = \big\{0, 8, 16, \ldots, 40\big\}\ .$$

The water volume in the dam at time t_i is known and equal to $x_i = 40$ hm^3, and the reference volume used to define the final payoff K in (5) is $x_{\text{ref}} = 40$ hm^3.

Prices scenario. The price scenario $\big(p(t_i), \ldots, p(t_f - 1)\big)$, known in advance, is shown in Figure 1.

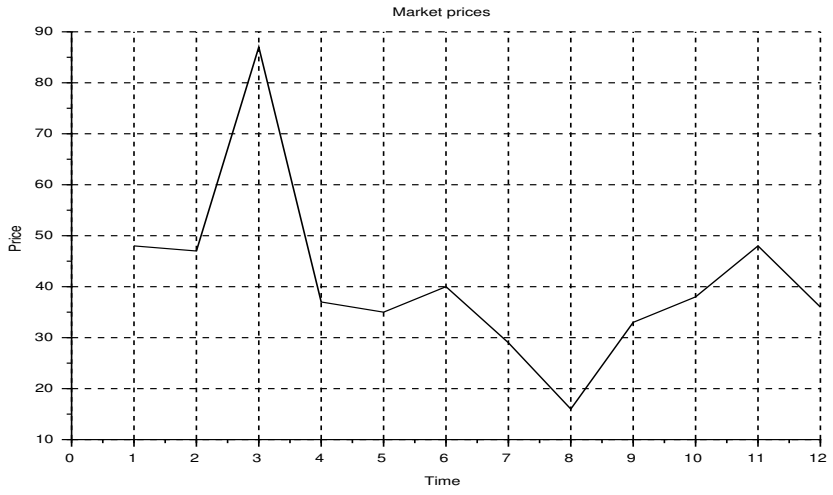

Figure 1: Prices trajectory for valuing the turbinated water

Water inflow scenarios. At each time t, the range $\mathbb{W}(t)$ of the water inflow $w(t)$ is obtained by discretizing an interval centered around a given mean value (see Figure 2). Each interval is discretized using a stepsize $\delta w = 2$ hm^3 (multiple of δx). Consider for example time $t = 1$: the minimum (resp. maximum) inflow value is $\underline{w}(1) = 12$ hm^3 (resp. $\overline{w}(1) = 28$ hm^3), so that the finite support $\mathbb{W}(1)$ of the (uniform) probability distribution associated with the random variable $w(1)$ is

$$\mathbb{W}(1) = \big\{12, 14, 16, \ldots, 28\big\} ,$$

the probability weight of each element $w \in \mathbb{W}(1)$ being $1/9$.

Knowing the probability distribution, it is easy to draw water inflow scenarios since we assumed that the random variables $w(t)$ are independent.

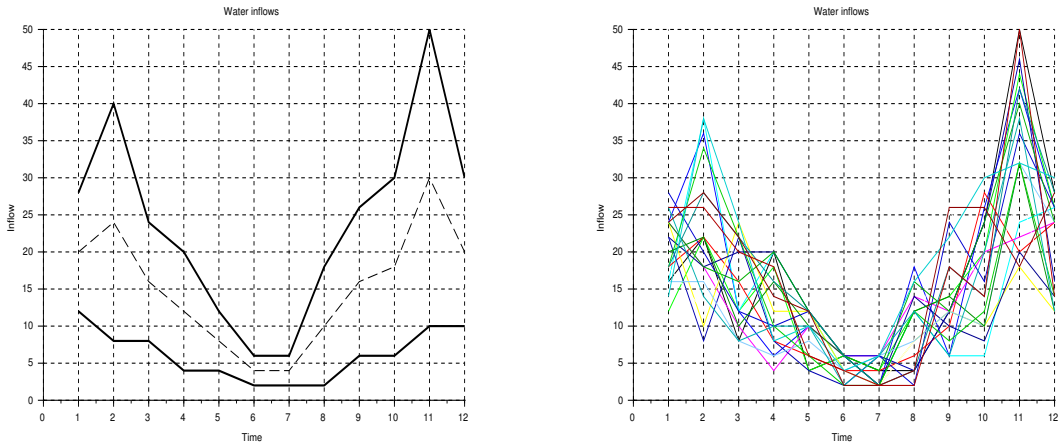

Figure 2: Inflows probability laws support (left) and some associated scenarios (right)

Scilab code. The following Scilab code contains the data and macros corresponding to the numerical example described in §1.4.

```
//-----------------------------
// PROBLEM DATA
//-----------------------------

// Time characteristics
// --------------------

// Time horizon
Ti=1;// Don't change this value!
Tf=13;
```

```
// Dam characteristics
// -------------------

xmin=0;
xmax=80;
xini=40;
xref=40;

// Dam discretization
xdlt=2;// Integer: xmin, xmax and xini are multiples of xdlt
Nx=((xmax-xmin)/xdlt)+1;

// Control characteristics
// -----------------------

umin=0;// Don't change this value!
umax=40;

// Control discretization
udlt=8;// Integer, multiple of xdlt: umin and umax are multiples of udlt
Nu=((umax-umin)/udlt)+1;

// Electricity prices
// ------------------

prices=[48.0,47.0,87.0,37.0,35.0,40.0,29.0,16.0,33.0,38.0,48.0,36.0];

// Inflows characteristics
// -----------------------

// Inflow discretization
wdlt=2;// Integer, multiple of xdlt

// Inflow mean values and maximal variations
wexp=[20.0,24.0,16.0,12.0,08.0,04.0,04.0,10.0,16.0,18.0,30.0,20.0];
wect=[08.0,16.0,08.0,08.0,04.0,02.0,02.0,08.0,10.0,12.0,20.0,10.0];

// Probability laws of the inflows (uniform law with finite support)
wlaws=list();
plaws=list();
for t=Ti:Tf-1 do
```

```
  wlaws(t)=[(wexp(t)-wect(t)):wdlt:(wexp(t)+wect(t))];
  plaws(t)=ones(wlaws(t))/length(wlaws(t));
end

// Macro generating Ns inflow scenarios
function wscenarios=inflow_scenarios(Ns,wlaws,plaws)
  // Random number generator
  grand("setgen","clcg2");
  grand("setsd",12345,67890);

  // Scenario generation
  wscenarios=zeros(Ns,Tf-1);
  for t=Ti:Tf-1 do
    // Probability law at t
    wlawt=wlaws(t);
    plawt=plaws(t);
    // Probability transition matrix for uniform sampling
    // We use grand(...,"markov",...,...) to generate independent sequences
    proba=ones(length(plawt),1)*plawt;
    samples=grand(Ns,"markov",proba,1);
    wscenarios(:,t)=wlawt(samples)';
  end
endfunction
```

Question 1

a) ***[1]*** *Copy this code into a file* ***DamManagement.sce*** *and check that it corresponds to the data given in §1.4.*

b) ***[2]*** *Launch Scilab and execute the file* ***DamManagement.sce****. Use the macro* ***inflow_scenarios*** *to generate 100 scenarios of water inflow. Plot the first 20 scenarios and compare them to the ones given in Figure 2.*

2 Simulation and evaluation of given policies

In order to simulate (and then optimize) the dam behavior, we need to have the Scilab macros computing the dynamics (1) and the payoffs (4) and (5).

```
//-------------------------------
// DAM MACROS
//-------------------------------
```

```
function xn=dynamics(t,x,u,w)
  // Dynamics of the dam
  xn=min(xmax,x-u+w);
endfunction

function payoff=instant_payoff(t,u)
  // Instantaneous payoff at time t
  payoff=prices(t)*u;
endfunction

function payoff=final_payoff(x)
  // Final payoff at horizon
  payoff=-min(0,x-xini) .^2;
endfunction
```

Question 2 *[2]* *Copy the above Scilab code at the end of the file* ***DamManagement.sce*** *and check that it corresponds to the expressions* (1)–(4)–(5)*. Notice that these macros are written in such a way that they may be fed either with scalar values or with vector values. This feature will be be used for efficiently simulating a bunch of inflow scenarios (see Question 3).*

2.1 Evaluation of a turbining policy

An *admissible policy* $\gamma : \mathbb{T} \times \mathbb{X} \to \mathbb{U}$ assigns a turbinated water amount $u = \gamma(t, x) \in \mathbb{U}$ to any time $t \in \mathbb{T}$ and to any dam volume $x \in \mathbb{X}$, while respecting constraint (3), that ias,

$$u \leq x \ .$$

Hence, by (2), we obtain that

$$u \leq \min\{x, \overline{u}\} \ .$$

Given an admissible policy γ and given an inflow scenario

$$w(\cdot) = \big(w(t_i), \ldots, w(t_f - 1)\big) \ , \tag{11}$$

we are able to build a dam volume trajectory

$$x(\cdot) := \big(x(t_i), \ldots, x(t_f - 1), x(t_f)\big) \tag{12}$$

and a turbinated water trajectory

$$u(\cdot) := \big(u(t_i), \ldots, u(t_f - 1)\big) \tag{13}$$

produced by the "closed-loop" dynamics initialized at the initial time t_i by

$$x(t_i) = x_i \tag{14a}$$

and propagated from $t = t_i$ up to $t = t_f - 1$ according to the policy

$$u(t) = \gamma\big(t, x(t)\big) \tag{14b}$$

and the dynamics (1)

$$x(t+1) = f\big(t, x(t), u(t), w(t+1)\big)\ . \tag{14c}$$

We also obtain the payoff associated to the inflow scenario $w(\cdot)$

$$J^\gamma\big(t_i, x_i, w(\cdot)\big) := \sum_{t=t_i}^{t_f-1} L\big(t, u(t)\big) + K\big(x(t_f)\big)\ , \tag{15}$$

where $x(\cdot)$ and $u(\cdot)$ are given by (14c). The *expected payoff* associated with the policy γ is

$$\mathbb{E}\Big(J^\gamma\big(t_i, x_i, w(\cdot)\big)\Big)\ , \tag{16}$$

where the expectation $\mathbb{E}$ is taken with respect to the product probability $\mathbb{P}$, whose marginals are given by (8). The true expected value (16) is difficult to compute,[3] and we evaluate it by the *Monte Carlo* method using N_s inflow scenarios $\big(w^1(\cdot), \dots, w^{N_s}(\cdot)\big)$:

$$\frac{1}{N_s}\sum_{s=1}^{N_s} J^\gamma\big(t_i, x_i, w^s(\cdot)\big)\ . \tag{17}$$

By the law of large numbers, the mean payoff (17) is a "good" approximation of the expected payoff(16) if the number of scenarios is "large enough".

We propose the following Scilab code in order to evaluate the mean payoff associated to a policy given by the Scilab macro `policy`.

```
//-----------------------------
// SCENARIO BASED SIMULATOR
//-----------------------------

function [xscenarios,uscenarios,cscenarios]=simulation(wscenarios,policy)
  // Initialization
  xscenarios=zeros(Ns,Tf);// used to store the state trajectories
  uscenarios=zeros(Ns,Tf-1);// used to store the control trajectories
  cscenarios=zeros(Ns);// used to store the payoff values

  // Simulation in forward time
  xscenarios(:,1)=xini*ones(Ns,1);
```

[3]Note however that this computation is achievable insofar all quantities it involves belong to finite sets.

```
  for t=Ti:Tf-1 do
    x=xscenarios(:,t);
    u=policy(t,x);
    w=wscenarios(:,t);
    xn=dynamics(t,x,u,w);
    cscenarios=cscenarios+instant_payoff(t,u);
    xscenarios(:,t+1)=xn;
    uscenarios(:,t)=u;
  end
  cscenarios=cscenarios+final_payoff(xn);
  printf('\n Mean payoff (Monte Carlo): %f\n',mean(cscenarios));
endfunction
```

Question 3

a) ***[1]*** *Copy the above Scilab code at the end of the file* ***DamManagement.sce****. Check that it corresponds to the expressions* (14c)–(15)–(17) *for a given policy* γ *(input argument* ***policy*** *of the macro* ***simulation****).*

b) ***[2]*** *Explain in detail how the macro* ***simulation*** *computes these quantities (see Question 2).*

2.2 Simulation of a given policy

In order to test the Scilab macro `simulation` given above, consider the following code.

```
//-------------------------------
// HEURISTIC POLICY EVALUATION
//-------------------------------

function u=heuristic_policy(t,x)
  // Heuristic policy
  thresmin=xref;
  thresmax=xref;
  u=(umax*bool2s(x > thresmin))+(umin*bool2s(x <= thresmax));
  u=min(u,x);
endfunction

// Scenarios generation
Ns=10000;
wscenarios=inflow_scenarios(Ns,wlaws,plaws);

// Simulation
```

```
[xscenarios,uscenarios,cscenarios]=simulation(wscenarios,heuristic_policy);

// Payoff histogram
xset("window",101);
clf();
histplot(50,cscenarios,normalization = %t,style = 5);
xgrid;
xtitle("Payoff distribution");

// State trajectories
xset("window",102);
clf();
plot2d([Ti:Tf],xscenarios(1:20,:)');
xgrid;
xtitle('Dam volume','Time','State');

// Control trajectories
xset("window",103);
clf();
plot2d([Ti:Tf-1],uscenarios(1:20,:)');
xgrid;
xtitle('Turbined water','Time','Control');
```

Question 4

(a) *[2]* *Explain the control policy induced by the Scilab macro* ***heuristic_policy****. Copy the relevant part of the above Scilab code at the end of* ***DamManagement.sce****, then generate 10,000 inflow scenarios and simulate the policy* ***heuristic_policy*** *along them.*

(b) *[2]* *Copy the relevant part of the above Scilab code at the end of* ***DamManagement.sce****, then plot some trajectories corresponding to the dam volume and to the turbinated water obtained using this policy, and provide a histogram of the payoff distribution. Comment the figures thus obtained.*

We now want to design new policies, writing Scilab macros following the model given by the macro below:

```
function u=my_policy(t,x)
  u=max(umin,min(WRITE_A_FORMULA_HERE,min(x,umax)));
endfunction
```

Question 5

(a) *[2]* *Explain why we use the expression* `max(umin,min( ..... , min(x,umax)))` *in the macro* ***my_policy*** *above.*

(b) ***[1+1+2]*** *Write the Scilab code of the following policies:*

- *always turbinate at the maximum possible (myopic),*
- *turbinate a fraction of the maximum possible,*

and evaluate them thanks to the scenario based simulator.

(c) ***[2]*** *Propose a policy which depends on the prices knowledge, and explain how you designed this policy.*

(d) ***[2]*** *Evaluate this last policy, and compare it to the ones designed in item (b).*

3 Resolution by the stochastic dynamic programming approach

In order to compute the optimal turbinated water policy, we provide the stochastic dynamic programming (or Bellman) equation associated with the problem of maximizing the expected payoff (16), as well as the corresponding Scilab code.

3.1 Stochastic dynamic programming equation

Since the water inflows $w(t)$ are supposed to be independent random variables, Dynamic Programming applies and the equation associated with the problem of *maximizing the expected payoff* (16) writes

$$\begin{aligned} V(t_f, x) &= K\big(x\big) \ , \\ V\big(t, x\big) &= \max_{u \in \mathbb{U} \ , \ u \leq x} \mathbb{E}\Big(L\big(t, u\big) + V\big(t+1, f(t, x, u, w(t+1))\big)\Big) \ , \end{aligned} \tag{18}$$

for t varying from $t_f - 1$ to t_i. The expectation $\mathbb{E}$ is taken with respect to the marginal probability distribution $\mathbb{P}_{w(t+1)}$ given by (8), that is,

$$\mathbb{E}\Big(L\big(t, u\big) + V\big(t+1, f(t, x, u, w(t+1))\big)\Big) = \frac{1}{\mathrm{card}\big(\mathbb{W}(\mathrm{t})\big)} \sum_{w \in \mathbb{W}(t)} L\big(t, u\big) + V\big(t+1, f(t, x, u, w)\big) \ .$$

A difficulty when programming the Bellman equation is that we need to manipulate both the *values* (t, x) of the time and of the state, and the associated *indices* (t, i) in the table storing the values of the function $V(t, x)$. In our application, the treatment of index t is obvious because $t \in \mathbb{T} = \{t_i, t_i + 1, \ldots, t_f\}$, but we need a tool performing the conversion for the volume. The following Scilab macro `state_index` returns the index i associated to an element x in the finite set $\mathbb{X}$ containing the possible values of the dam volume.

```
function i=state_index(x)
  // Index associated with a volume value
  i=dsearch(x,[xmin:xdlt:xmax],'d');
  if i==0 then
    printf('\n Invalid dam volume: %f\n',x);
  end
endfunction
```

Question 6 *[2]* *Copy the above Scilab code at the end of the file* `DamManagement.sce`. *Explain in detail how the macro works.*

3.2 Scilab code for the dynamic programming equation

We provide the following code in order to compute the Bellman function.

```
//-------------------------------
// STOCHASTIC DYNAMIC PROGRAMMING
//-------------------------------

// Initialization
Vbell=-ones(Tf,Nx)*%inf;// Used to store the optimal payoff functions
Ubell=ones(Tf-1,Nx)*%inf;// Used to store the optimal controls

// SDP computation by backward induction
function [Vbell,Ubell]=sdp_solve()
  // Bellman function at final time
  payoff=final_payoff([xmin:xdlt:xmax]);
  Vbell(Tf,:)=payoff';
  // Loop backward in time
  for t=Tf-1:-1:Ti do
    wlawt=wlaws(t);
    plawt=plaws(t);
    Nw=length(wlawt);
    for x=xmin:xdlt:xmax do
      i=state_index(x);
      for u=[umin:udlt:min(x,umax)] do
        ctot=0;
        for k=1:Nw do
          w=wlawt(k);
          p=plawt(k);
          xn=dynamics(t,x,u,w);
          in=state_index(xn);
          ctot=ctot+(p*Vbell(t+1,in));
```

```
        end
        ctot=ctot+instant_payoff(t,u);
        if ctot > Vbell(t,i) then
          Vbell(t,i)=ctot;
          Ubell(t,i)=u;
        end
      end
    end
  end
endfunction
```

Question 7

(a) ***[2]*** *Copy the above Scilab code at the end of the file* ***DamManagement.sce****. Explain in detail why the macro* ***sdp_solve*** *corresponds to the computation of the Dynamic Programming equation* (18)*.*

(b) ***[1]*** *Execute the macro* ***sdp_solve****, which generates the table* ***Vbell*** *containing the values of the Bellman function and the table* ***Ubell*** *containing the values of the optimal controls. Display the value of the Bellman function at* (t_i, x_i)*.*

(c) ***[2]*** *Plot different Bellman functions and comment on their shape.*

3.3 Simulation of the optimal policy

Consider the following Scilab macro.

```
function u=optimal_policy(t,x)
  i=state_index(x);
  u=Ubell(t,i)';
endfunction
```

Question 8

(a) ***[2]*** *Explain why the control policy induced by the Scilab macro* ***optimal_policy*** *is optimal.*

(b) ***[1]*** *Generate 10,000 water inflow scenarios using the macro* ***simulation*** *and simulate the policy* ***optimal_policy*** *along them. Plot some trajectories corresponding to the dam volume and to the turbinated water, obtained using this policy.*

(c) ***[2]*** *Provide a histogram of the distribution of the random payoff, and compare the mean payoff* (17)*, with the value* $V(t_i, x_i)$ *of the Bellman function evaluated at* (t_i, x_i)*.*

(d) ***[3]*** *Compare these results with those obtained using all previous policies.*

Here we assume that the values of optimal control *have not be stored* in the table `Ubell`, so that the only outcome of the macro `sdp_solve` is the table `Vbell` (Bellman values).

Question 9 *[2] Provide a method and the corresponding code to compute the optimal value of the control at time t for a given dam volume $x(t)$ using only the values stored in the table `Vbell` at time $t+1$.*

Hint: *the optimal control at $(t, x(t))$ is the* $\arg\max$ *of the following optimization problem:*

$$\max_{u \in \mathbb{U}\ ,\ u \leq x(t)} \mathbb{E}\Big(L\big(t,u\big) + V\big(t+1, f(t,x(t),u,w)\big)\Big)\ . \tag{19}$$

Be careful that your code must accept a vector of volumes as input.

4 Possible extensions

In this section, we propose several developments and extensions for the dam management problem.

4.1 Hazard-decision framework

In the *hazard-decision* framework, the dam manager is supposed to make a decision, here turbining $u(t)$ at the beginning of the period $[t, t+1[$, *knowing* the water inflow $w(t+1)$ in advance.

The Bellman equation is now

$$\begin{aligned} V(t_f, x) &= K\big(x\big) \ , \\ V\big(t, x\big) &= \mathbb{E}\Big(\max_{u \in \mathbb{U} \ , \ u \leq x} L\big(t, u\big) + V\big(t+1, f(t, x, u, w(t+1))\big)\Big) \ . \end{aligned} \tag{20}$$

Once the value function $V(t, x)$ evaluated and stored for all time t and state x, the optimal control at time t starting from state x and observing the water inflow w is the arg max of

$$\max_{u \in \mathbb{U} \ , \ u \leq x} L\big(t, u\big) + V\big(t+1, f(t, x, u, w)\big) \ , \tag{21}$$

which leads to an optimal policy depending on the triplet (t, x, w).

Question 10

(a) ***[1]*** *Explain the difference between* (20) *and* (18).

(b) ***[2]*** *Adapt the Scilab macro* `sdp_solve` *to solve the hazard-decision Bellman equation* (20). *We* decide not to store *the optimal policy in the table* `Ubell`.

(c) ***[1+2]*** *Based on* (21)*, write the Scilab code which computes the optimal control as a function of* (t, x, w)*, and use it to obtain by simulation the optimal trajectories of the state and the control as well as the optimal payoff distribution. Compare.*

4.2 Computation of a fair final value of water

Till now, the gain in leaving water in the dam at the end of the time horizon was arbitrarily fixed. Now, we provide a procedure to estimate a " fair final value of water", that is, a function $x \mapsto K(x)$ as in (5).

The intuition behind the procedure is that the final value of water is the value that a manager would put on the dam, were he to run it from the date $t_f + 1$ to infinity. The final value of water is the solution of an infinite horizon maximization problem. The procedure below mimicks an algorithm to find a fixed point by iterations.

First, we start with a zero final payoff function $K^{(1)}(\cdot) \equiv 0$ and obtain, by backward induction, the Bellman function $V^{(1)}(t_i, \cdot)$ at initial time t_i. Up to a translation — to

account for the fact that an empty dam has zero final value of water — we identify the function $V^{(1)}(t_i, \cdot) - V^{(1)}(t_i, \underline{x})$ with the new final value of water $K^{(2)}(\cdot)$. Proceeding along, we expect that this loop converges towards a function $x \mapsto K^{(\infty)}(x)$ which is a good candidate for the final value of water.

We design a loop for $n = 1, \ldots, N$, starting with a zero final value of water

$$K^{(1)}(x) = 0\,, \quad \forall x \in \mathbb{X}\,,$$

then solving the backward Bellman equation

$$\begin{aligned} V^{(n)}(t_f, x) &= K^{(n)}\big(x\big)\,, \\ V^{(n)}\big(t, x\big) &= \max_{u \in \mathbb{U}\,,\, u \leq x} \mathbb{E}\Big(L\big(t, u\big) + V^{(n)}\big(t+1, f(t, x, u, w(t+1))\big)\Big)\,, \end{aligned}$$

and then closing the loop by choosing the new final value of water

$$K^{(n+1)}(x) = V^{(n)}(t_i, x) - V^{(n)}(t_i, \underline{x})\,, \quad \forall x \in \mathbb{X}\,, \tag{22}$$

as the Bellman function at the initial time t_i after a shift, so that $K^{(n+1)}(\underline{x}) = 0$.

Question 11 *Implement the iterative procedure described above, and compute and plot the "fair final value of water" in the dam.*

4.3 Robustness with respect to water inflow scenarios

An interesting point, which occurs in practical problems, is the following. In the case where the random variables $\big(w(t_i), \ldots, w(t_f - 1)\big)$ are *correlated in time*, it is always possible to obtain the marginal probability laws $\mathbb{P}_{w(t)}$ of each $w(t)$ and then compute the Bellman function as in §3. We thus obtain a feedback policy that would be optimal if the random inflow variables were uncorrelated. The loss of optimality may be evaluated by simulating the feedback policy along scenarios drawn using the "true" probability law $\mathbb{P}$ of the process $\big(w(t_i), \ldots, w(t_f - 1)\big)$.

Question 12

(a) Draw inflow scenarios according to an auto-regressive AR(1) process.

(b) Compute the marginal probability distribution $\mathbb{P}_{w(t)}$ induced by these scenarios.

(c) Compute the Bellman solution of the problem using these marginal probability laws and the associated optimal policy.

(d) Using this policy, run a simulation along scenarios drawn according to the AR(1) process.

(e) Evaluate the loss of optimality induced by ignoring the correlation in the inflow process.

References

D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, Massachusets, second edition, 2000. Volumes 1 and 2.

M. De Lara and L. Doyen. *Sustainable Management of Natural Resources. Mathematical Models and Methods*. Springer-Verlag, Berlin, 2008.