Dam Optimal and Viable Deterministic Management

Michel De Lara

October 10, 2017

Contents

Abstract

We consider a dam manager intenting to maximize the intertemporal payoff obtained by selling hydropower produced by water releases, when the water inflows (rain, outflows from upper dams) and the energy prices are supposed to be deterministic. You will propose different water turbined policies and you will compute their payoff. Then, you will compute the optimal payoff and display the optimal policy. At last, you will evaluate the cost of introducing a "tourism" constraint consisting in having a guaranteed stock volume in the dam during Summer months.

1 Problem statement and data

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

1.1 Dam dynamics

$$\underbrace{S(t+1)}_{\text{future volume}} = \min\{S^\sharp, \underbrace{S(t)}_{\text{volume}} - \underbrace{q(t)}_{\text{turbined}} + \underbrace{a(t)}_{\text{inflow}}\}\ , \quad t \in \mathbb{T} := \{t_0, \dots, T-1\} \tag{1}$$

with

- time $t \in \overline{\mathbb{T}} := \{t_0, \dots, T\}$ is discrete (days), and t denotes the beginning of the period $[t, t+1[$; we call t_0 the *initial time* and T the *horizon*;
- $S(t)$ *volume* (stock) of water at the beginning of period $[t, t+1[$, belonging to the discrete set $\mathbb{X} = \{0, 1, 2, \dots, S^\sharp\}$, made of water volumes (h m^3), where $S^\sharp$ is the maximum dam volume;
- $a(t)$ *inflow water volume* (rain, etc.) during $[t, t+1[$, belonging to $\mathbb{W} = \{0, 1, 2, \dots, a^\sharp\}$, supposed to be deterministic, hence known in advance at initial time t_0;
- $q(t)$ *turbined outflow volume* during $[t, t+1[$, decided at the beginning of period $[t, t+1[$, belonging to the discrete set $\mathbb{U} = \{0, 1, 2, \dots, q^\sharp\}$, where $q^\sharp$ is the maximum which can be turbined by time unit (and produce electricity), and such that

$$0 \leq q(t) \leq S(t)\ . \tag{2}$$

A *scenario* of water inputs

$$a(\cdot) := \big(a(t_0), \dots, a(T-1)\big) \tag{3}$$

is given (data of the problem), hence known in advance at initial time t_0 where the following optimization problem is posed.

1.2 Criterion: intertemporal payoff

The manager original problem is one of *payoff maximization* where turbining one unit of water has unitary *price* $p(t)$. On the period from t_0 to T, the payoffs sum up to

$$\sum_{t=t_0}^{T-1} p(t)q(t) + \mathtt{K}\big(S(T)\big)\ , \tag{4}$$

where

- a *scenario* of prices
$$p(\cdot) = \big(p(t_0), \dots, p(T-1)\big) \tag{5}$$
is given (data of the problem) and supposed to be known in advance at initial time t_0 where the optimization problem is posed (deterministic setting);

- the final term $\mathtt{K}\big(S(T)\big)$ is called the *final value of water*, since it values the water volume $S(T)$ in the dam at the horizon T.

1.3 Numerical data

Time. We consider a daily management (the interval $[t, t+1[$ represents one day) over one year
$$t_0 = 1 \qquad \text{and} \qquad T = 365\ . \tag{6}$$

Bounds. Concerning the dam and water inflows, we consider the following bounds:
$$S_0 = 0\ \text{hm}^3\ , \quad S^\sharp = 100\ \text{hm}^3\ , \quad q^\sharp = \frac{0.4}{7} \times S^\sharp \qquad \text{and} \qquad a^\sharp = \frac{0.5}{7} \times S^\sharp\ . \tag{7}$$
These figures reflect the assumptions that, during one week, one can release at maximum 40% of the dam volume, and that during one week of full water inflows, an empty dam can be half-filled.

Prices scenario. The scenario $p(\cdot) = \big(p(t_0), \dots, p(T-1)\big)$ of prices is known in advance. We produce it by one sample from the expression
$$p(t) = \big(1 + \frac{1}{4}\sin(\frac{2\pi t}{T} + \frac{\pi}{2}) + \epsilon(t)\big) \times \overline{p} \qquad \text{with} \qquad \overline{p} = 66\text{MWh/hm}^3 \times 2.7\ \text{euros/MWh}^3 \tag{8}$$
where $\epsilon(t)$ is drawn from a sequence of i.i.d. uniform random variables in $[-0.1, 0.1]$. The above expression reflects the assumption that prices are seasonal (high in winter for house heating, and low in summer).

Water inflows scenario. The scenario $a(\cdot) := \big(a(t_0), \dots, a(T-1)\big)$ of water inflows is known in advance. We produce it by one sample from the expression
$$a(t) = \text{closest integer to } \frac{1}{3}\big(1 + \sin(\frac{2\pi t}{T} + \frac{\pi}{2}) + \eta(t)\big) \times a^\sharp \tag{9}$$
where $\eta(t)$ is drawn from a sequence of i.i.d. uniform random variables in $[0, 1]$. The above expression reflects the assumption that water inflows are seasonal (high in winter and low in summer).

Copy the following Scicoslab code into a file `DamData.sce`. Check that it indeed corresponds to the data above.

```
// exec DamData.sce
// ----------------------------------------------------
// DATA
// ----------------------------------------------------

// State
volume_max=100;
volume_min=0;

// Control
control_max=round(0.4/7*volume_max);

// Time
tt0=1;
horizon=365;
TT=tt0:(horizon-1);
bTT=tt0:(horizon);

// Prices
price=66*2.7*(1+1/4*sin(2*%pi*TT/horizon+%pi/2)+1/5*(rand(TT)-1/2));

// Water inflows
inflow_max=floor(0.5/7*volume_max);
Scenario=inflow_max*1/3*(1+sin(2*%pi*bTT/horizon+%pi/2)+rand(bTT));
Scenario=round(Scenario);

xset("window",11);xbasc();
plot2d2(bTT,Scenario')
```

2 Simulation and evaluation of given policies

First, we detail how a water turbined policy yields state (stock volume) and control (turbined) trajectories by the dynamics (1), that are then evaluated with the criterion (4). Second, you will propose different policies and you will compute their payoff.

2.1 Evaluation of a water turbined policy

An *admissible policy* $\gamma : \mathbb{T} \times \mathbb{X} \to \mathbb{U}$ assigns a water turbined $q = \gamma(t, S)$ to any state S of dam stock volume and to any decision period $t \in \mathbb{T}$, while respecting the constraints

$$q \in \{0, 1, 2, \ldots, q^\sharp\} \qquad \text{and} \qquad 0 \leq q \leq S \,, \tag{10}$$

that is,

$$q \in \{0, 1, 2, \ldots, \min\{S, q^\sharp\}\} \; . \tag{11}$$

Once given, we obtain a volume trajectory $S(\cdot) := \big(S(t_0), \ldots, S(T-1), S(T)\big)$ and a turbined trajectory $q(\cdot) := \big(q(t_0), \ldots, q(T-1)\big)$ produced by the "closed-loop" dynamics

$$\begin{array}{rcl} S(t_0) & = & S_0 \\ S(t+1) & = & \min\{S^\sharp, S(t) - q(t) + a(t)\} \\ q(t) & = & \gamma\big(t, S(t)\big) \end{array} \tag{12}$$

and function of the scenario $a(\cdot)$ of water inputs in (3). Thus, in the end, we obtain the payoff

$$\mathtt{J}^\gamma\big(t_0, S_0\big) := \sum_{t=t_0}^{T-1} p(t)q(t) + \mathtt{K}\big(S(T)\big) \; , \tag{13}$$

as a function of the policy γ, where the trajectories $S(\cdot)$ and $q(\cdot)$ are given by (12).

Copy the following Scicoslab code into the file `DamData.sce`. Check that it indeed corresponds to the expressions (12) and (13) in the special case where the final gain $\mathtt{K} = 0$.

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

// Dynamics
function ssdot=dynamics(ss,qq,aa)
  ssdot=max(volume_min,min(volume_max,ss-qq+aa));
endfunction

// Instantaneous payoff function
function cc=instant_payoff(tt,ss,qq,aa)
  cc=price(tt)*qq;
endfunction

// Final payoff function
function cc=final_payoff(ss)
  cc=0;
endfunction


// Trajectories simulations

function [SS,QQ,CC]=trajectories(ss0,policy,scenarios)
  SS=[];
```

```
  QQ=[];
  CC=[];

  nb_simulations=size(scenarios,'r');

  for kksimu=1:nb_simulations do
    ss=ss0;
    qq=[];
    cc=0;
    aa=scenarios(kksimu,:);

    for tt=TT do
      qq=[qq,policy(tt,ss($))];
      ss=[ss,dynamics(ss($),qq($),aa(tt))];
      cc=cc+instant_payoff(tt,ss($),qq($),aa(tt));
    end
    cc=cc+final_payoff(ss($));

    SS=[SS;ss];
    QQ=[QQ;qq];
    CC=[CC;cc];
  end
  //
  disp('The payoff given by simulation is '+string(CC));
endfunction
```

2.2 Numerical evaluation of the payoff of different policies

Load the file `DamData.sce` by the instruction

```
// load data
exec DamData.sce
```

Copy the following Scicoslab code into a new file `DamOptimality.sce`. The code allows you to answer to Question 1.

```
// exec DamOptimality.sce
// ----------------------------------------------------
//  SIMULATIONS
// ----------------------------------------------------

// Example of policy
function qq=half_policy(tt,ss)
```

```
  qq=maxi(0,min(ss/2,min(ss,control_max)));
endfunction

ss0=0;
policy=half_policy;
scenarios=Scenario;

// Trajectories simulations and visualization
[SS,QQ,CC]=trajectories(ss0,policy,scenarios);
xset("window",21);xbasc();
plot2d2(TT,[SS(1:($-1));QQ]')
xtitle('Stock and turbined volumes in a dam','(time)','(volume)')
```

Question 1

(a) Picture the trajectory of the stocks and of the turbined volumes corresponding to the policy $\gamma(t,S) = \min\{S/2, q^\sharp\}$ *— which turbines half of the stock, except if it is higher than the capacity* $q^\sharp$ *— and evaluate the payoff as in* (4).

(b) Explain what the macro **`trajectories`** *is calculating.*

Now, you will design new policies, writing code as below, following the example of the `half_policy` policy.

```
function qq=my_policy(tt,ss)
  qq=maxi(0,min("Write A Formula Here",min(ss,control_max)));
endfunction
```

Question 2

(a) Explain the expression `maxi(0,min( write a formula here , min(ss,control_max)))` *above.*

(b) Following the above example **`my_policy`**, *give the Scicoslab code of other policies: always turbinate at the maximum possible (myopic), turbinate a fraction of the maximum possible.*

(c) Picture the prices trajectory. Propose two policies which depend on the scenario of prices (do not use "if... then..." loops, but expressions like `bool2s(price(tt)>mean(price))`*). Explain how you designed these two policies.*

(d) Evaluate, and display in a table, the payoffs associated with the above policies starting from initial empty stock $S_0 = 0$ *at initial time* t_0*. What is the best policy among them?*

3 Dynamic programming equation

As a step towards computing an optimal water turbined policy, we provide the dynamic programming (or Bellman) equation associated with the problem of maximizing the payoff (13), as well as the corresponding Scicoslab code.

3.1 Additive dynamic programming equation

The dynamic programming equation associated with the problem of maximizing the payoff (13) is

$$\begin{aligned} V(T,S) &= \overbrace{\mathtt{K}(S)}^{\text{final payoff}} \ , \\ V(t,S) &= \max_{q\in\{0,1,2,\ldots,\min\{S,q^\sharp\}\}} \left(\underbrace{p(t)q}_{\text{instant. payoff}} + V\big(t+1, \underbrace{\min\{S^\sharp, S-q+a(t)\}}_{\text{future stock volume}}\big) \right) \ . \end{aligned} \tag{14}$$

Recall the definition of the *characteristic function* χ_A of a set A: $\chi_A(q) = 0$ if $q \in A$, $\chi_A(q) = +\infty$ if $q \notin A$. Getting rid of the constraint $q \in \{0,1,2,\ldots,\min\{S,q^\sharp\}\}$ by incorporating it in the corresponding characteristic function, the dynamic programming equation above can be written under the equivalent form

$$\begin{aligned} V(T,S) &= \mathtt{K}(S) \ , \\ V(t,S) &= \max_q \big(-\chi_{\{0,1,\ldots,\min\{S,q^\sharp\}\}}(q) + p(t)q + V\big(t+1, \min\{S^\sharp, S-q+a(t)\}\big)\big) \ . \end{aligned} \tag{15}$$

This trick will be used in the Scicoslab code. We will replace the characteristic function χ_A by a penalization: $\bar{\chi}_A(q) = 0$ if $q \in A$, $\bar{\chi}_A(q) = 1/\epsilon$ if $q \notin A$, where ϵ is a very small positive number.

3.2 Scicoslab code for the additive dynamic programming equation

Copy the following Scicoslab code into the file `DamOptimality.sce`. Check that it indeed corresponds to the Bellman equation (15).

```
////////////////////////////////////////////////////////////////////
//    ADDITIVE DYNAMIC PROGRAMMING EQUATION
////////////////////////////////////////////////////////////////////

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

```
states=[0:volume_max];
controls=[0:control_max];

cardinal_states=size(states,'c');
cardinal_controls=size(controls,'c');

state_min=min(states);
state_max=max(states);

// ---------------------------------------------------
//  MACROS
// ---------------------------------------------------
function [FEEDBACK,VALUE]=DDP(FINAL_PAYOFF_VECTOR)
  VALUE=zeros(bTT'*states);
  FEEDBACK=zeros(TT'*states);

  VALUE(horizon,:)=FINAL_PAYOFF_VECTOR;// vector

  // backward time
  for tt=TT($:-1:1) do
    loc=zeros(cardinal_controls,cardinal_states);
    // local variable containg the values of the function to be optimized
    for jj=1:cardinal_controls do
      qq=controls(jj);
      penalty=-1/%eps*bool2s(states < qq);
      // penalization of the control constraint
      loc(jj,:)=0;
      aa=Scenario(tt);
      loc(jj,:)=loc(jj,:)+penalty+instant_payoff(tt,states,qq,aa)+ ...
                VALUE(tt+1,dynamics(states,qq,aa)-state_min+1);
      // with an approximation -1/%eps of -infinity
    end
    //
    [mmn,jjn]=min(loc,'r');
    [mmx,jjx]=max(loc,'r');
    // mm is the extremum achieved
    // jj is the index of the extremum argument
    //
    VALUE(tt,:)=mmx;
    // maximal payoff
    FEEDBACK(tt,:)=controls(jjx);
    // optimal feedback
```

```
  end
endfunction
```

4 Optimal policies

First, we will go on using a zero final value of water, meaning that leaving a stock in the dam at the end of the year is not valued. Thanks to the Scicoslab code for the Bellman equation (15), you will compute the optimal payoff and compare it with the payoffs obtained in the previous questions. Second, we will display a procedure to give proper value to a final stock. You will then compute the optimal policy and study it. At last, you will examine if this policy, designed for a single specific water inflows scenario, is sensitive to a change of scenario (robustness).

4.1 Optimal policy with a zero final value of water

We first consider that the final "value of water" is zero:

$$\mathtt{K}(S) = 0\,, \quad \forall S \in \mathbb{X}\,. \tag{16}$$

Copy the following Scicoslab code into the file `DamOptimality.sce`.

```
// We start with a zero value of water at the end of the year
zero_final_payoff_vector=zeros(states);

// -----------------------------------------------------
//  SIMULATIONS
// -----------------------------------------------------
[FEEDBACK,VALUE]=DDP(zero_final_payoff_vector);

// optimal policy
function uu=zero_optimal_policy(tt,ss)
  uu=FEEDBACK(tt,ss-state_min+1);
  // shift due to the difference between
  // volume indices and volume physical values
endfunction

ss0=0;
policy=zero_optimal_policy;
scenarios=Scenario;

// Trajectories simulations and visualization
[SS,QQ,CC]=trajectories(ss0,policy,scenarios);
xset("window",31);xbasc();
```

```
plot2d(bTT,SS')
xtitle('Stock volumes in a dam following an optimal policy '+ ...
       'with a zero final value of water','(time)','(volume)')

xset("window",32);xbasc();
plot2d2(TT,[SS(1:($-1));QQ]')
xtitle('Stock and turbined volumes in a dam following an optimal policy '+ ...
       'with a zero final value of water','(time)','(volume)')

disp('The optimal payoff given by simulation is '+string(CC));

disp('The optimal payoff given by the Bellman function is '+string(VALUE(tt0,1)));
```

Question 3

(a) What is the numerical value of the Bellman function $V(t_0, 0)$ evaluated at initial time t_0 and initial empty stock $S_0 = 0$?

(b) Making connections with the course, recall what is $V(t_0, 0)$ by definition.

(c) Scrutinize the matrix ***FEEDBACK*** *in the macro* ***DDP*** *and how* ***FEEDBACK*** *is used in the macro* ***zero_optimal_policy****. Explain why the macro* ***zero_optimal_policy*** *corresponds to what has been called* optimal policy *in the course.*

(d) What is the value of the payoff along the trajectory of stocks and turbinated obtained with the optimal policy given by the Bellman equation? This value should be equal to $V(t_0, 0)$.

(e) Explain why these two values are equal. For this purpose, you will make connections with the course. You will use, in a proper way, the following basic theoretical result in dynamic programming: the Bellman equation yields a policy which is optimal in that it produces the highest payoff. (This has nothing to do with the final value of water being zero, nor with the fact that the Bellman equation is backward).

(f) Picture the trajectory of the stocks corresponding to the optimal policy. What do you observe for the final stock and for the final turbinated decision? Explain why.

4.2 Computation of the final value of water

Till now, there was no gain in leaving water in the dam at the ultimate decision period as reflected in (16). Now, we consider that the final gain `K` is non zero, and we provide a procedure to estimate a "final value of water", that is, a function $S \mapsto \mathtt{K}(S)$ as in (4).

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 $T + 1$ 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 value of water $\mathtt{K}^{(1)}(S) = 0$ and obtain, by backward induction, the Bellman function $V^{(1)}(t_0, S)$ at initial time. Up to a translation — to account for the fact that an empty dam has zero final value of water — we identify $V^{(1)}(t_0, S) - V^{(1)}(t_0, 0)$ with the new final value of water $\mathtt{K}^{(2)}(S)$. Proceeding along, we expect that this loop converges towards a function $S \mapsto \mathtt{K}^{(\infty)}(S)$ which is a good candidate for the final value of water.

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

$$\mathtt{K}^{(1)}(S) = 0\,, \quad \forall S \in \mathbb{X} \tag{17}$$

then solving the backward Bellman equation

$$\begin{array}{rcl} V^{(k)}(T, S) & = & \mathtt{K}^{(k)}(S)\,, \\ V^{(k)}(t, S) & = & \displaystyle\max_{q \in \{0,1,2,\ldots,\min\{S,q^\sharp\}\}} \big(p(t)q + V^{(k)}\big(t+1, \min\{S^\sharp, S - q + a(t)\}\big)\big)\,, \end{array} \tag{18}$$

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

$$\mathtt{K}^{(k+1)}(S) = V^{(k)}(t_0, S) - V^{(k)}(t_0, 0)\,, \quad \forall S \in \mathbb{X}\,, \tag{19}$$

as the Bellman function at the initial time t_0 after translation, so that $\mathtt{K}^{(k+1)}(0) = 0$.

Copy the following Scicoslab code into the file `DamOptimality.sce`.

```
////////////////////////////////////////////////////////////////////////
//    FINAL VALUE OF WATER
////////////////////////////////////////////////////////////////////////

// We start with a zero value of water at the end of the year
zero_final_payoff_vector=zeros(states);

[FEEDBACK,VALUE]=DDP(zero_final_payoff_vector);

// Then, we set the initial value VALUE(1,:)-VALUE(1,1)
// obtained as the new final value of water
// VALUE(1,1) is the initial Bellman function for a zero stock

for k=1:5 do
  final_payoff_vector=VALUE(1,:)-VALUE(1,1);
  // disp('The minimum of the initial value is '+string(min(VALUE(1,:))));
  // disp('At step '+string(k) +...
  //', the mean of the final value is '+string(mean(VALUE(1,:))));
  //disp('The maximum of the initial value is '+string(max(VALUE(1,:))));
  [FEEDBACK,VALUE]=DDP(final_payoff_vector);
  disp('At step '+string(k)+ ...
```

```
        ', the norm of the difference between successive final values is '+ ...
        string(norm(final_payoff_vector-(VALUE(1,:)-VALUE(1,1)))));
end

xset("window",41);xbasc();
plot2d(states,final_payoff_vector)
xtitle('Final value of water','(volume)','(euros)')
```

Question 4

(a) In how many iterations do you observe that the loop converges?

(b) Display the final value of water as a function of the volume in the dam, thanks to the vector ***final_payoff_vector*** *which contains the sequence of value of water for each volume in the dam. How does the final value of water depend upon the volume? Is it increasing, linear, convex, concave, etc.?*

4.3 Optimal policy with a non zero final value of water

Copy the following Scicoslab code into the file `DamOptimality.sce`.

```
// ---------------------------------------------------
//  SIMULATIONS
// ---------------------------------------------------

[FEEDBACK,VALUE]=DDP(final_payoff_vector);
// optimal policy
function uu=optimal_policy(tt,xx)
  uu=FEEDBACK(tt,xx-state_min+1);
endfunction

// Final payoff function
function c=final_payoff(ss)
  c=final_payoff_vector(ss+1);
endfunction

ss0=0;
policy=optimal_policy;
scenarios=Scenario;

// Trajectories simulations and visualization
[SS,QQ,CC]=trajectories(ss0,policy,scenarios);
xset("window",42);xbasc();
plot2d(bTT,SS')
```

```
xtitle('Stock volumes in a dam following an optimal policy '+'with a final value of wate
       '(time)','(volume)')

disp('The optimal payoff given by the Bellman function is '+string(VALUE(tt0,1)));

// Randomly selected times
snapshots=grand(1,5,'uin',TT(1),TT($));

for tt=snapshots do
  number_window=4000+tt;
  xset("window",number_window);xbasc();
  plot2d2(states,optimal_policy(tt,states))
  xtitle('Optimal policy at time '+string(tt),'(volume)','(turbined)')
end
```

Question 5

(a) Draw curves $S \mapsto q = \gamma^\star(t, S)$ for some values of time t, where $\gamma^\star$ is the optimal policy numerically found (use ***plot2d2****). Comment on their forms and on the structure of optimal policies.*

(b) Picture the trajectory of the stocks corresponding to the optimal policy. What do you observe for the final stock and for the final turbinated decision? Explain why.

(c) Evaluate the optimal payoffs for different values of the initial stock S_0. Give the values of the Bellman function $V(t_0, S_0)$ evaluated at initial time t_0 and the same values of the initial stock S_0. Write them all in a table and compare them.

4.4 Robustness with respect to water inflows scenarios

The optimal policy discussed in Question 5 is taylored for one single inflows scenario. We now want to figure out how it responds to other inflows scenarios.

Question 6

(a) Draw 250 water inflows scenarios, and run 250 simulations, starting from an empty dam ($S_0 = 0$), with the optimal policy of Question 5. Picture the histogram of the 250 payoffs.

(b) Comment on the location of the optimal payoff found at Question 5 with the range of possible values when the inflows scenario changes.

```
nbsimu=250;
SCENARIOS=zeros(nbsimu,1)*zeros(bTT);
scenarios=ones(nbsimu,1)*inflow_max*1/3* ...
```

```
          (1+sin(2*%pi*ones(bTT)/horizon+%pi/2)+rand(SCENARIOS));
ss0=0;
policy=optimal_policy;
// Trajectories simulations and visualization
[SS,QQ,CC]=trajectories(ss0,policy,scenarios);
xset("window",43);xbasc();
histplot(10,CC)
```

5 Optimal and viable management with guaranteed minimal volume during Summer months

For "tourism" reasons, the following *viability* constraint is imposed on the stock volume during some Summer months:

$$S^\flat \leq S(t) \ , \quad \forall t \in \{\text{July}, \text{August}\} \ . \tag{20}$$

Such a state constraint can be treated by dynamic programming by incorporating it in the corresponding characteristic function.

Question 7

(a) Change the above Scicoslab code to account for the "tourism" viability constraint (20) *with* $S^\flat = 70\% \times S^\sharp$.

(b) Picture the trajectory of the stocks corresponding to the optimal policy.

(c) Evaluate the optimal payoff, and compare it with the value function $V(t_0, 0)$ *evaluated at the initial time* t_0 *and the initial stock* $S_0 = 0$.

```
summer_state=0.7*volume_max;

function [FEEDBACK,VALUE]=DDP(FINAL_PAYOFF_VECTOR)
  VALUE=zeros(bTT'*states);
  FEEDBACK=zeros(TT'*states);

  VALUE(horizon,:)=FINAL_PAYOFF_VECTOR;// vector

  // backward time
  for tt=TT($:-1:1) do
    loc=zeros(cardinal_controls,cardinal_states);
    // local variable containg the values of the function to be optimized
    for jj=1:cardinal_controls do
      qq=controls(jj);
```

```
      penalty=-1/%eps*bool2s(states < qq)+ ...
              -1/%eps*bool2s(tt > 152 & tt < 243)*bool2s(states < summer_state);
      // bool2s(states<qq)
      //       is 1 iff the control constraint is NOT satisfied
      // bool2s(tt>152 & tt<243) * bool2s(states<summer_state)
      //       is 1 iff the state constraint is NOT satisfied
      loc(jj,:)=0;
      aa=Scenario(tt);
      loc(jj,:)=loc(jj,:)+penalty+instant_payoff(tt,states,qq,aa)+ ...
                VALUE(tt+1,dynamics(states,qq,aa)-state_min+1);
      // with an approximation -1/%eps of -infinity
    end
    //
    [mmn,jjn]=min(loc,'r');
    [mmx,jjx]=max(loc,'r');
    // mm is the extremum achieved
    // jj is the index of the extremum argument
    //
    VALUE(tt,:)=mmx;
    // maximal payoff
    FEEDBACK(tt,:)=controls(jjx);
    // optimal feedback
  end
endfunction
```

The viability constraint (20) is an additional constraint, hence reduces the domain of optimization. Therefore, the payoff is lower.

Question 8 *For $S^\flat$ varying regularly between 0% to 100% of $S^\sharp$, compute the "cost of the viability constraint", as the difference between the optimal payoff without and with the "tourism" viability constraint.*

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.