Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Extended Ritz method for reservoir management over an infinite horizon

Francesca Pianosi * **Rodolfo Soncini-Sessa** **

* *Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan, Italy (Tel: +39-2-2399.9630; e-mail: pianosi@elet.polimi.it).*
** *Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan, Italy (Tel: +39-2-2399.3551; e-mail: soncini@elet.polimi.it).*

Abstract: The management problem of water reservoirs can be formulated as a stochastic optimal control (SOC) problem, where the objective function is an aggregated cost that accounts for the interests acting in the water system (e.g. hydropower production, irrigation supply, etc.) and the design variable is the reservoirs release policy. Solving the SOC problem through stochastic dynamic programming is often impossible, since the numerical resolution of the Bellman equation is computationally prohibitive even for small reservoir networks. An approximate solution can be searched for by assuming a priori the family of function to which the control laws belong and replacing the SOC problem with a nonlinear programming one. Recently, a method based on this approach has been proposed in the literature, coupled with the use of nonlinear approximating networks to approximate the optimal control laws. This optimization method, called Extended RItz Method (ERIM), is suited for finite horizon SOC problems. However, management problems for environmental systems are spontaneously formulated over an infinite horizon, since the life time of the system is infinite. This paper thus presents an extension of the ERIM to the infinite horizon case. The algorithm that implements such method is tested on a numerical example where a 10-reservoirs network is optimized for hydropower production and irrigation supply.

1. INTRODUCTION

The policy design problem for a reservoir network can be formulated as a Stochastic Optimal Control (SOC) problem, where the objective function is obtained by aggregating a number of partial cost functions that express the different interests in play in the water system (e.g. flood control, hydropower production, irrigation supply, ecosystem conservation). Since the water system is dynamical, with uncertain inputs (namely inflow to the reservoirs) and possibly constrained on the state and/or on the control, Stochastic Dynamic Programming (SDP) appears to be the most suitable solution approach to the SOC problem. Unfortunately, the analytical solution obtained in the so-called LQG framework can not be exploited in most of the real world applications, since none of the assumptions of the LQG framework (linear system, quadratic objective function and Gaussian random inputs) is satisfied. An approximate solution can be obtained by discretizing the state, control and disturbance spaces, and numerically solving the Bellman equation. The limit of this approach is that its computing time increases exponentially with the number of components of the state, control and disturbance vectors, thus making the problem intractable even for 'simple' water systems, e.g. a few reservoirs in the network (*curse of dimensionality*, Bellman (1957)). In order to overcome this difficulty, many approaches have been proposed. Some are based on a manipulation of the problem aimed at making it tractable with SDP, e.g. by simplifying the model of the water system or using smart approximators for the cost-to-go functions or reducing the discretization grid giving up regular grids for the state space discretization and using Montecarlo or quasi-Montecarlo. Others abandon SDP and turn to different optimization techniques: either choosing a priori the family of functions to which the control laws belong, thus transforming the SOC problem into a mathematical programming one, or by using a partial model-free approach and reinforcement learning algorithms. Recent reviews of the above approaches, as well as discussion of strengths and difficulties of each method can be found in Baglietto et al. (2006) and Soncini-Sessa et al. (2007).

This paper focuses on the Extended RItz Method (ERIM) proposed by Zoppoli et al. (2002). Here, an Approximating Network (AN) is used as a fixed-class control law and its parameters are optimized based on the Ritz method. This method has been developed for finite horizon SOC problems. Unfortunately, when managing environmental systems, the assumption that the life-time of the system be finite is unrealistic. A finite horizon could be considered only if it were possible to define a penalty function over the ('fake') final state, to express the cost from that instant on. By doing so, however, the problem of dealing with an infinite horizon is just shifted to the definition of the penalty function. In the case of real-time management, the algorithm developed for the finite horizon case can be exploited also over an infinite horizon, through the application of the receding horizon principle. However, again the problem arises of choosing the appropriate penalty

* Corresponding author F. Pianosi. Tel. +39-2-2399.9630. Fax +39-2-2399.9611.

function over the final state of the receding horizon: unappropriate choice, in fact, can compromise the performances of the control scheme.
Following the above considerations, in this paper an extension of the ERIM to the case of infinite horizon SOC problems is proposed. The paper is organized as follows. In the next section, the management problem for a reservoir network is formulated as a SOC problem for either the finite and infinite horizon case. In section 3, the SOC problem is transformed into a nonlinear programming problem by introducing suitable approximators for the optimal control laws. The optimization of the approximator parameters is based on a gradient descent method, thus requiring the knowledge of the gradient of the cost function. The approximate computation of the latter is the keystone for the extension of the ERIM to the infinite horizon case. Therefore, we will first introduce a new method for the computation of the gradient over a finite horizon; this is equivalent to the one proposed by Zoppoli et al. (2002), however it has the advantage that it can be easily extended to the infinite horizon case, as it will be shown in section 3.3. The proposed approach is tested (section 4) on a simplified 10-reservoirs system that is often used as a test system in the literature of water systems management. Finally, a brief discussion of the open issues of the proposed approach is given in the last section.

2. PROBLEM STATEMENT

The water system is composed of reservoirs, natural catchments that feed the reservoirs, diversion dams, water users (e.g. hydropower plants or irrigation districts) and artificial and natural canals that connect all the above components. It can be described as a discrete-time dynamical system. Discrete time is considered because the decision time step is discrete: release decisions are usually taken daily and, in any case, at least every few hours, because of physical constraints in the implementation of the decision (e.g., operating the dam's gates). The system dynamics is given by the following time-varying state transition equation

$$x_{t+1} = f_t(x_t, u_t, \varepsilon_{t+1}) \tag{1}$$

where $x_t \in \mathbb{R}^{n_x}$ and $u_t \in U_t \subseteq \mathbb{R}^{n_u}$ are the state and control at time instant t; and $\varepsilon_{t+1} \in \mathbb{R}^{n_\varepsilon}$ is the disturbance acting in the time interval $[t, t+1)$, which is generated by a white noise process. In the adopted notation, the time subscript of a variable indicates the instant when the its value is deterministically known. The state x_t is composed of the state variables of the reservoirs, i.e. their storages, and, when the case, the state variables of the catchments, of the canals and of the water users. The control is composed of the release decisions of the reservoirs and the distribution decisions at the regulated diversion dams, if any. The disturbance ε_{t+1} is a random vector and as such, at each time t, it is described by its probability density function (pdf) $\phi_t(\cdot)$.
The global performances of the system over the finite horizon $[0, h]$ can be evaluated by means of the following cost function

$$J = \underset{\varepsilon_1^h}{E}\left[\sum_{t=0}^{h-1} g_t(x_t, u_t, \varepsilon_{t+1}) + g_h(x_h)\right] \tag{2}$$

where the notation ε_1^h is used to indicate the trajectory of the variable ε_t from time 1 to time h, $g_t(\cdot)$ is a scalar function that expresses the step-cost associated to the system transition from t to $t+1$ and $g_h(\cdot)$ is the penalty function over the final state. The step-cost and penalty functions are derived by linear combination of the partial step-costs and penalties that express the costs incurred by the single water users', e.g. irrigation deficit, flooded area, etc. (for the meaning of the aggregation in the perspective of multi-objective optimization, see Soncini-Sessa et al. (2007) and references therein).
If, at each time t, the control is taken on the basis of a control law $u_t = m_t(x_t)$, the finite horizon SOC problem can be formulated as:

Problem P1 Find the sequence $\{u_t^* = m_t^*(x_t) \in U_t : t = 0, \ldots, h-1\}$ of optimal control laws that minimizes (2) for a given initial state x_0.

2.1 Formulation of the infinite-horizon SOC problem

If an infinite horizon is considered, the cost function cannot be defined as in equation (2). In fact, first the penalty function $g_h(\cdot)$ is no more necessary. Furthermore, some corrections must be introduced to avoid the divergence of J. To this end, two approaches are possible. The first consists in introducing a discount factor γ (with $0 < \gamma < 1$) and defining J as the expected Total Discounted Cost (TDC), i.e.

$$J = \lim_{h\to\infty} \underset{\varepsilon_1^h}{E}\left[\sum_{t=0}^{h-1} \gamma^t g_t(x_t, u_t, \varepsilon_{t+1})\right] \tag{3}$$

The second consists in considering the Average Expected Value (AEV)

$$J = \lim_{h\to\infty} \underset{\varepsilon_1^h}{E}\left[\frac{1}{h}\sum_{t=0}^{h-1} g_t(x_t, u_t, \varepsilon_{t+1})\right] \tag{4}$$

The infinite horizon SOC problem is now well posed and it can be solved provided that the water system be cyclostationary, i.e. the functions $f_t(\cdot)$ in (1), $g_t(\cdot)$ in (3)-(4) and the disturbance pdf $\phi_t(\cdot)$ be periodic. This is common in water systems, where the period T is usually equal to one year. Under this assumption, the infinite horizon SOC problem is formulated as:

Problem P2 Find the periodic sequence $\{u_t^* = m_t^*(x_t) \in U_t : t = 0, 1, \ldots; m_t^*(\cdot) = m_{t+T}^*(\cdot)\}$ of optimal control laws that minimizes (3) - or (4) - for a given initial state x_0.

3. EXTENDED RITZ METHOD

The difficulty in solving the SOC problem formulated in previous section (both in its finite and infinite version) stems from the fact that each control law $m_t^*(\cdot)$ belongs to an infinite-dimensional space of functions. The problem is simplified if, for each $t = 0, 1, \ldots$, the control law is forced to belong to a pre-selected family of functions $\{\hat{m}_t(x_t, \theta_t) : \theta_t \in \Theta_t\}$, where $\Theta_t \subseteq \mathbb{R}^{n_\theta}$ is such that $\hat{m}_t(x_t, \theta_t) \in U_t$ for any x_t. Then, the cost function J is a function of the sequence $\{\theta_t : t = 1, 2, \ldots\}$ and the SOC problem turns into a nonlinear programming one. For example, in the finite horizon case, the SOC problem P1 is replaced by the following

Problem P3 Find the sequence $\{\theta_t^* : t = 0, \ldots, h-1\}$ of optimal parameters that minimizes (3) - or (4) - with $u_t^* = \hat{m}_t(x_t, \theta_t^*)$ for each t, $\hat{m}_t(\cdot)$ belonging to the a priori selected family $\{\hat{m}_t(\cdot, \theta_t) : \theta_t \in \Theta_t\}$, and given initial state x_0.

3.1 Solution of the nonlinear programming problem

In general, problem P3 is just an approximation of the original SOC problem P1, because the optimal control laws $m_t^*(\cdot)$ that compose the solution of the latter might not belong to the pre-selected families $\{\hat{m}_t(\cdot, \theta_t)\}$. Zoppoli et al. (2002) discuss the very concept of 'approximation' of an optimization problem and provide results about the accuracy of the approximate solution $\{\hat{m}_t(\cdot, \theta_t^*) : t = 0, \ldots, h-1\}$ when the functions $\hat{m}_t(\cdot, \theta_t)$ are Approximating Networks (ANs). An Approximating Network is a nonlinear function obtained as a linear combination of simple basis functions. More precisely, if u_t is obtained as the output of an AN, its j-th component is given by

$$u_t^j(x_t) = \sum_{i=1}^{\nu} c_{ij} \psi_i(x_t, k_i) \tag{5}$$

where $\psi_i(\cdot)$, i =1,...,ν are basis functions and $k_i \in \mathbb{R}^k$ are inner parameters. The parameter vector that collects all the AN parameters is $\theta_t = \mathrm{col}(c_{ij}, k_i : i = 1, \ldots, \nu; j = 1, \ldots, n_u)$ and it belongs to $\mathbb{R}^{n_\theta}$ with $n_\theta = \nu(k + n_u)$. In the following, for the sake of simplicity and without loss of generality, it will be assumed that the ANs $\hat{m}_t(\cdot)$ all have the same structure (same number and type of basis functions) for $t = 0$, ..., $h-1$. If no inner parameters k_i appear in the basis functions, the ANs are linear in the unknown parameters c_{ij} and Problem P3 can be solved by means of the classical Ritz method.

Zoppoli et al. (2002) propose an extension of the Ritz method for solving problem P3 when nonlinear ANs are used, i.e. the parameters k_i are to be determined too, and call it Extended RItz Method (ERIM). They also propose an algorithm based on gradient descent for solving problem P3 when the parameters are unconstrained ($\Theta_t = \mathbb{R}^{n_\theta}$). Conventional gradient descent method foresees that the estimate of the parameter vector $\theta = \mathrm{col}(\theta_t : t = 0, \ldots, h-1)$ be iteratively derived as

$$\theta^{i+1} = \theta^i - \alpha_i \nabla_\theta J \tag{6}$$

However, this would require the computation of the gradient of J. Analytical computation is almost always impossible because of the complexity of the definition of J, and numerical computation is time consuming.

In order to reduce the computational effort, the gradient $\nabla_\theta J$ in (6) can be replaced by the gradient $\nabla_\theta Z$, where Z is the total cost associated to a single realization of the disturbance,

$$Z = \sum_{t=0}^{h-1} g_t(x_t, u_t, \varepsilon_{t+1}) + g_h(x_h) \tag{7}$$

The algorithm so obtained belongs to the class of stochastic approximation algorithms (see e.g. Kushner and Yin (1997)). In order to guarantee convergence of the algorithm, it is necessary that the step-size α_i tends to zero as i goes to infinity; here, the form $\alpha_i = c_1/(c_2 + i)$ will be considered.

The gradient $\nabla_\theta Z$ can be computed rather efficiently: each partial derivative $\partial Z/\partial \theta_t^j$ (j =1,...,n_θ) that compose $\nabla_\theta Z$ can be obtained from the product $\partial Z/\partial u_t \cdot \partial \hat{m}_k/\partial \theta_t^j$, while $\partial Z/\partial u_t$ is obtained through an algebraic combination of $\partial g_t/\partial u_t$, $\partial f_t/\partial u_t$ and $\partial Z/\partial x_{t+1}$; and $\partial Z/\partial x_t$ is recursively computed, backward in time, based on $\partial g_t/\partial x_t$, $\partial f_t/\partial x_t$, $\partial \hat{m}_t/\partial x_t$ and $\partial Z/\partial u_t$ (see for example Zoppoli et al. (2002)).

3.2 Alternative computation of the gradient

The backward computation of $\partial Z/\partial x_t$ presented in previous paragraph requires an initialization that is possible only in the finite horizon case, where $\partial Z/\partial x_t = \partial g_h/\partial x_h$. In the infinite horizon case, instead, the final instant goes to infinity and $\partial Z/\partial x_t$ can not be initialized. Therefore, in order to extend the method to problems defined over an infinite horizon, it is first necessary to derive an algorithm for the forward computation of $\nabla_\theta Z$. This is easy: in fact, from (7) it follows that each of the hn_θ partial derivatives $\partial Z/\partial \theta_t^j$, t=0,...,$h-1$ and j=1,...,n_θ, is given by the following equation

$$\frac{\partial Z}{\partial \theta_t^j} = \frac{\partial}{\partial \theta_t^j}\left(\sum_{k=0}^{h-1} g_k(x_k, u_k, \varepsilon_{k+1}) + g_h(x_h)\right) =$$

$$= \frac{\partial g_t}{\partial u_t}\frac{\partial \hat{m}_t}{\partial \theta_t^j} + \sum_{k=t+1}^{h-1}\left(\frac{\partial g_k}{\partial x_k} + \frac{\partial g_k}{\partial u_k}\frac{\partial \hat{m}_k}{\partial x_k}\right)\frac{\partial x_k}{\partial \theta_t^j} + \frac{\partial g_h}{\partial x_h}\frac{\partial x_h}{\partial \theta_t^j}$$

The sensitivity vector $\partial x_k/\partial \theta_t^j$ (k=t+1,..., h) that appears in the above equation can be recursively obtained by posing $\partial x_t/\partial \theta_t^j = 0$ and using the following equation (sensitivity system) for $k = t, \ldots, h-1$

$$\frac{\partial x_{k+1}}{\partial \theta_t^j} = \frac{\partial f_k}{\partial x_k}\frac{\partial x_k}{\partial \theta_t^j} + \frac{\partial f_k}{\partial u_k}\left(\frac{\partial \hat{m}_k}{\partial \theta_t^j} + \frac{\partial \hat{m}_k}{\partial x_k}\frac{\partial x_k}{\partial \theta_t^j}\right) =$$

$$= \left(\frac{\partial f_k}{\partial x_k} + \frac{\partial f_k}{\partial u_k}\frac{\partial \hat{m}_k}{\partial x_k}\right)\frac{\partial x_k}{\partial \theta_t^j} + \frac{\partial f_k}{\partial u_k}\frac{\partial \hat{m}_k}{\partial \theta_t^j} \tag{8}$$

Note that the term $\partial \hat{m}_k/\partial \theta_t^j$ is zero for any k exception made for $k = t$; therefore the second term on the right hand side of equation (8) actually appears only for $k = t$.

It is easy to prove that the backward and forward computation of the gradient are equivalent. However, the backward computation is preferable when the ANs $\hat{m}_t$ are multilayer neural networks. In this case, in fact, the computation of the partial derivatives $\partial \hat{m}_t/\partial \theta_t^j$ and $\partial \hat{m}_t/\partial x_t$ can require some tedious algebra, which can be avoided if the backward procedure is used. In fact, with little modification the backward procedure can be combined with the back-propagation equations, in order to directly compute the partial derivatives $\partial Z/\partial \theta_t^j$ without requiring the knowledge of $\partial \hat{m}_t/\partial \theta_t^j$ and $\partial \hat{m}_t/\partial x_t$. The same cannot be done with the forward procedure. However, the forward procedure was proposed because, to the author's knowledge, it is the only one that can be applied in the infinite horizon case, as it shall be seen in the next section.

3.3 Extended Ritz method for the infinite-horizon problem

In the infinite horizon case, we consider a cyclostationary nonlinear system of period T and search for a periodic

policy, i.e. a periodic sequence of control laws $m_t^*(\cdot)$ (for the study of the optimality properties of periodic policies for infinite horizon SOC problems, see Bertsekas (1976)). Following the approximate approach introduced in previous section, we search for a periodic sequence of approximators $\hat{m}_t(\cdot,\theta^*)$, i.e. we formulate the following nonlinear programming problem.

Problem P4 Find the periodic sequence $\{\theta_t^* : t = 0,1,\ldots;\theta_t^* = \theta_{t+T}^*\}$ of optimal parameters that minimizes (3) - or (4) - with $u_t^* = \hat{m}_t(x_t,\theta_t^*)$ for each t, $\hat{m}_t(\cdot)$ belonging to the a priori selected family $\{\hat{m}_t(\cdot,\theta_t) : \theta_t \in \Theta_t\}$, and given initial state x_0.

This problem can be solved by using the same approach as in the finite horizon case, i.e. a gradient descent method where $\nabla_\theta J$ is replaced by $\nabla_\theta Z$. The parameter vector is $\theta = \mathrm{col}(\theta_t : t = 0,\ldots,T-1)$ and it has $n_\theta T$ components, n_θ being the number of components of each $\theta_t, t = 0,\ldots,T-1$. The proposed stochastic approximation algorithm is

$$\theta^{i+1} = \theta^i - \alpha_i \nabla_\theta Z \tag{9}$$

where the total cost Z is defined as

$$Z = \sum_{t=0}^{h-1} \gamma^t g_t(x_t,u_t,\varepsilon_{t+1}) \tag{10}$$

if the TDC formulation is used, or

$$Z = \frac{1}{h}\sum_{t=0}^{h-1} g_t(x_t,u_t,\varepsilon_{t+1}) \tag{11}$$

if the AEV formulation is used. The algorithm works as follows: at each iteration, the gradient $\nabla_\theta Z$ is evaluated at the current parameter estimate and subject to a randomly extracted trajectory $\bar{\varepsilon}_1^h$ of the disturbance, where the finite length h of the horizon is suitably chosen (we shall go back to this issue in the following). Once h has been fixed, the computation of $\nabla_\theta Z$ is straightforward, as it will be shown in what follows. Before proceeding, however, it is interesting to note that, under the assumption that the operations of limit and derivation can be exchanged, the gradient $\nabla_\theta Z$ in (9) can be viewed as an approximation of $\nabla_\theta J$ where (a) the expected cost is replaced by the cost subject to a particular disturbance trajectory; and (b) the cost is computed based on the trajectories of the system variables truncated at time h instead of the entire trajectories over an infinite horizon.

Back to the computation of $\nabla_\theta Z$, the forward computation introduced in sec. 3.2 will be exploited. In fact, if Z is defined as in equation (10), its Tn_θ partial derivatives $\partial Z/\partial\theta_t^j$ (for t=0,...,T−1 and j =1,...,n_θ) are given by

$$\frac{\partial Z}{\partial\theta_t^j} = \sum_{k=t}^{h-1}\gamma^k\left[\frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial\theta_t^j} + \left(\frac{\partial g_k}{\partial x_k} + \frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial x_k}\right)\frac{\partial x_k}{\partial\theta_t^j}\right] \tag{12a}$$

while, if Z is defined as in equation (11), they are given by

$$\frac{\partial Z}{\partial\theta_t^j} = \frac{1}{h}\sum_{k=t}^{h-1}\left[\frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial\theta_t^j} + \left(\frac{\partial g_k}{\partial x_k} + \frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial x_k}\right)\frac{\partial x_k}{\partial\theta_t^j}\right] \tag{12b}$$

The sensitivity vector $\partial x_k/\partial\theta_t^j$, for k=t,t+1,... can be obtained by posing $\partial x_t/\partial\theta_t^j = 0$ and recursively using equation (8). Note that here, since we consider a periodic policy, $\partial\hat{m}_k/\partial\theta_t^j = \partial\hat{m}_t/\partial\theta_t^j$ if k=t+nT ($n \in \mathbb{N}$) and zero otherwise.

Now, the question arises on how to choose the parameter h that appears in equations (12a) and (12b). To obtain a reasonable choice of h, it is first necessary to derive a recursive formula for the computation of the partial derivatives $\partial Z/\partial\theta_t^j$. To this end, let us denote by z_{tj}^τ the partial summations

$$z_{tj}^\tau = \sum_{k=t}^{\tau}\gamma^k\left[\frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial\theta_t^j} + \left(\frac{\partial g_k}{\partial x_k} + \frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial x_k}\right)\frac{\partial x_k}{\partial\theta_t^j}\right]$$

in the TDC case, and

$$z_{tj}^\tau = \frac{1}{\tau+1}\sum_{k=t}^{\tau}\left[\frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial\theta_t^j} + \left(\frac{\partial g_k}{\partial x_k} + \frac{\partial g_k}{\partial u_k}\frac{\partial\hat{m}_k}{\partial x_k}\right)\frac{\partial x_k}{\partial\theta_t^j}\right]$$

in the AEV case. Then, the partial derivatives that compose $\nabla_\theta Z$ are given by

$$\frac{\partial Z}{\partial\theta_t^j} = z_{tj}^{h-1}$$

for t=0,...,T-1 and j=1,...,n_θ. The partial derivatives z_{tj}^{h-1} can be obtained by recursively computing z_{tj}^τ for τ=0,...,h-1. To this end, notice that from equations (12a) and (12b) it follows that, for $t = 0,\ldots,T-1$ and $j = 1,\ldots,n_\theta$,

$$z_{tj}^\tau = \begin{cases} 0 & \text{if } \tau < t \\ z_{tj}^{\tau-1} + \gamma^\tau\left[\dfrac{\partial g_\tau}{\partial u_\tau}\dfrac{\partial\hat{m}_\tau}{\partial\theta_t^j} + \right. & \\ \qquad \left. + \left(\dfrac{\partial g_\tau}{\partial x_\tau} + \dfrac{\partial g_\tau}{\partial u_\tau}\dfrac{\partial\hat{m}_\tau}{\partial x_\tau}\right)\dfrac{\partial x_\tau}{\partial\theta_t^j}\right] & \text{if } \tau \geq t \end{cases} \tag{13a}$$

in the TDC case and

$$z_{tj}^\tau = \begin{cases} 0 & \text{if } \tau < t \\ \dfrac{\tau-1}{\tau}z_{tj}^{\tau-1} + \dfrac{1}{\tau}\left[\dfrac{\partial g_\tau}{\partial u_\tau}\dfrac{\partial\hat{m}_\tau}{\partial\theta_t^j}\right. & \\ \qquad \left. + \left(\dfrac{\partial g_\tau}{\partial x_\tau} + \dfrac{\partial g_\tau}{\partial u_\tau}\dfrac{\partial\hat{m}_\tau}{\partial x_\tau}\right)\dfrac{\partial x_\tau}{\partial\theta_t^j}\right] & \text{if } \tau \geq t \end{cases} \tag{13b}$$

in the AEV case. In the next paragraph, the above equations will be used to derive a reasonable value of h. The same equations are exploited in the algorithm for the computation of the gradient $\nabla_\theta Z$, which is as follows.

Algorithm for the computation of $\nabla_\theta Z$

Initialization

Choose the length h of the simulation horizon and randomly extract a trajectory $\bar{\varepsilon}_1^h$.
Set the state x_0 to the given initial state value and let $\tau = 0$.

Iteration. While $\tau < h$:

Compute the control value $u_\tau = \hat{m}_\tau(x_\tau,\theta_t)$, with $t = \mathrm{mod}(\tau,T)$, based on the current estimate θ^i of the parameter vector.

For $t = 1,\ldots,T-1$ and $j = 1,\ldots,n_\theta$:
- compute z_{tj}^τ with (13a) or (13b)
- compute the state of the sensitivity system $\partial x_{\tau+1}/\partial\theta_t^j$

according to (8) where k is replaced by τ
(all the partial derivatives are evaluated at point $(x_\tau, u_\tau, \bar{\varepsilon}_{\tau+1}, \theta^i)$)

Compute the state $x_{\tau+1} = f_\tau(x_\tau, u_\tau, \bar{\varepsilon}_{\tau+1})$.
Increase τ of one unity.

Termination

The algorithm terminates when $\tau = h$. The values z_{tj}^{h-1}, for $t = 1, \ldots, T-1$ and $j = 1, \ldots, n_\theta$, are the partial derivatives $\partial Z / \partial \theta_t^j$ that compose the gradient $\nabla_\theta Z$ in (6).

3.4 Choice of the length of the simulation horizon

From equations (13a) and (13b) it can be seen that, if the terms in brackets are finite for any τ and any $t = 0, \ldots, T-1, j = 1, \ldots, n_\theta$, then $z_{tj}^\tau \simeq z_{tj}^{\tau-1}$ for τ sufficiently large. This correspond to saying that the limit $\lim_{h\to\infty} \nabla_\theta Z$ exists and is finite. Therefore, a criterion for selecting h may be that of setting it to the time instant at which the change in the value of all the z_{tj}^τ becomes negligible. The choice of h can thus be made once and for all before starting the stochastic approximation algorithm, by proceeding as follows. Let $\bar{\tau}$ be a positive scalar such that

$$|z_{tj}^\tau - z_{tj}^{\tau-1}| \leq \alpha_Z \quad \forall \tau \geq \bar{\tau} \text{ and } \forall t, j$$

where α_Z is a prefixed accuracy value. Once α_Z is fixed, the value of $\bar{\tau}$ can be derived by solving one of the following equalities

$$\gamma^\tau \bar{g} = \alpha_Z \quad \text{or} \quad \frac{1}{\tau}\bar{g} = \alpha_Z$$

where $\bar{g}$ is the maximum possible value of the second term on the right hand side of equations (13a) and (13b), i.e.

$$\bar{g} = \max_{x_\tau, u_\tau, \varepsilon_{\tau+1}, \tau, \theta} \left| \frac{\partial g_\tau}{\partial u_\tau} \frac{\partial \hat{m}_\tau}{\partial \theta_t^j} + \left(\frac{\partial g_\tau}{\partial x_\tau} + \frac{\partial g_\tau}{\partial u_\tau} \frac{\partial \hat{m}_\tau}{\partial x_\tau} \right) X_{tj}^\tau \right|$$

Then, the length of the finite horizon h can be set to $\bar{\tau}$. With this choice, the truncated summation z_{ij}^h turns out to be an estimate of $\lim_{h\to\infty} \partial Z / \partial \theta_i^j$ with accuracy α_Z.

However, the recursive computation of z_{ij}^τ might be also arrested at a time $h < \bar{\tau}$, thus reducing the computing time for deriving $\nabla_\theta Z$. The effect of different choices of h have been empirically studied on the case study presented in section 4, which showed that satisfactory optimization results can be obtained also for relatively small h (of the order of $100T$). However, the choice of such parameter and the analysis of the effects of this choice remains an open issue of the presented approach.

4. NUMERICAL EXAMPLE

The proposed algorithm has been tested on the 10-reservoir system presented in Yakowitz (1982) and in Baglietto et al. (2006). The system topology is shown in figure 1. The state transition function of the j-th reservoir (j=1,...,10) is

$$x_{t+1}^j = \min \left(x_t^j - u_t^j + \sum_{h \in I_j^+} u_t^h + e_{t+1}^j, x_{\max}^j \right) \quad (14a)$$

$$0 \leq u_t^j \leq s_t^j + \sum_{h \in I_j^+} u_t^h + e_{\min}^j \quad (14b)$$

where x_t^j is the water storage in the j-th reservoir at time t and $x_{\max}^j$ its maximum storage, u_t^j is the release decision from the j-th reservoir, I_j^+ denotes the set of indexes of the reservoirs that release water directly into the j-th one, and e_{t+1}^j is the inflow to the j-th reservoir from the uncontrolled catchment in the time interval $[t, t+1)$. In model (14), superficial spills that occur when the storage overcomes its maximum value $x_{\max}^j$ are assumed not to reach the downstream reservoirs. The natural inflow e_{t+1}^j is described as a stochastic variable drawn from a continuous uniform distribution on the interval $[e_{\min}^j, e_{\max}^j]$, and is assumed to be independent of other inflows e_{t+1}^i, $i \neq j$. In Baglietto et al. (2006), the interval $[e_{\min}^j, e_{\max}^j]$ is the same for all t, here we assume that the boundaries $e_{\min}^j$ and $e_{\max}^j$ be a periodic function of time, thus making the water system time-variant and periodic. The time period T has been assumed equal to 12. The benefit function to be maximized is of the form (3) - or (4) - with a step-cost function defined as

$$g_t(x_t, u_t, \varepsilon_{t+1}) = g(u_t) = c_p \cdot u_t + c_f u_t^{10} \quad (15)$$

where the product $c_p \cdot u_t$ accounts for the benefit from hydropower generation downstream of each reservoir and $c_f u_t^{10}$ accounts for the benefit from irrigation downstream of the 10th reservoir.
A nonlinear approximating network is used to approximate the optimal control law. The release decision u_t^j from the jth reservoir is computed as a function of the state vector x_t=col$(x_t^j : j = 1, ..., 10)$ as in equation (5) with

$$\psi(x_t, k_i) = \frac{1}{1 + \exp(-x_t' \cdot \alpha_i + \beta_i)}$$

where k_i=col$(\alpha_i, \beta_i : i = 1, ..., \nu)$, α_i being a parameter vector of the same size as x_t. When estimating the control law parameter with the algorithm presented in the previous section, suitable penalty function have been added to (15) in order to implement the constraint (14b). The system parameter and initial state where set to the values reported in Baglietto et al. (2006), except for the minimum and maximum natural inflows, for which T values were considered by randomly perturbing the original values proposed in Baglietto et al. (2006) (see table 1).

Several optimization experiments were run, with both TDC and AEV formulations and different number ν of basis functions in the nonlinear AN (5). In each experiment, the initial value θ^0 of the parameter vector was randomly extracted from a uniform distribution over $[0, 0.005]$. The length h other simulation horizon used for the computation of $\nabla_\theta Z$ was set to 1200. Results were compared through Monte Carlo simulation of the system, subject to the optimized policies, over a horizon of again 1200 steps (see table 2). Ten realizations of the disturbance trajectory were considered in the Monte Carlo simulation. As it can be noticed from the table, the performances of the system are significantly improved after optimization of the control

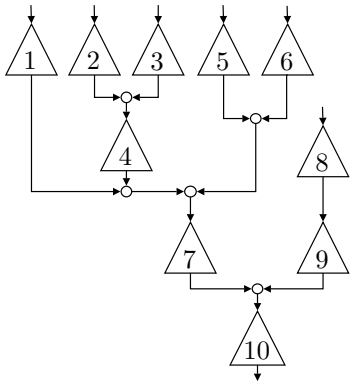

Fig. 1. The configuration of the 10-reservoir system considered in the numerical example.

Table 1. Parameter and initial state values used in the numerical example.

j	1	2	3	4	5	6	7	8	9	10	
$x^j_{\max}$	10	10	10	10	10	10	10	10	18	25	
x^j_0	10	10	10	10	10	10	10	10	18	25	
$e^j_{\min}$	0.98	0.98	0.98	0	0.68	0.68	0	1.47	0	0	($T=1$
	0.94	0.94	0.94	0	0.65	0.65	0	1.40	0	0	2
	1.09	1.09	1.09	0	0.76	0.76	0	1.63	0	0	3
	1.01	1.01	1.01	0	0.71	0.71	0	1.52	0	0	4
	0.94	0.94	0.94	0	0.66	0.66	0	1.41	0	0	5
	0.96	0.96	0.96	0	0.67	0.67	0	1.44	0	0	6
	1.00	1.00	1.00	0	0.70	0.70	0	1.50	0	0	7
	1.07	1.07	1.07	0	0.75	0.75	0	1.60	0	0	8
	0.97	0.97	0.97	0	0.68	0.68	0	1.45	0	0	9
	0.98	0.98	0.98	0	0.69	0.69	0	1.47	0	0	10
	1.04	1.04	1.04	0	0.73	0.73	0	1.56	0	0	11
	0.97	0.97	0.97	0	0.68	0.68	0	1.45	0	0	12)
$e^j_{\max}$	1.47	1.47	1.47	0	1.17	1.17	0	1.95	0	0	($T=1$
	1.40	1.40	1.40	0	1.12	1.12	0	1.87	0	0	2
	1.63	1.63	1.63	0	1.31	1.31	0	2.18	0	0	3
	1.52	1.52	1.52	0	1.22	1.22	0	2.03	0	0	4
	1.41	1.41	1.41	0	1.13	1.13	0	1.88	0	0	5
	1.44	1.44	1.44	0	1.15	1.15	0	1.92	0	0	6
	1.50	1.50	1.50	0	1.20	1.20	0	2.01	0	0	7
	1.60	1.60	1.60	0	1.28	1.28	0	2.14	0	0	8
	1.45	1.45	1.45	0	1.16	1.16	0	1.94	0	0	9
	1.47	1.47	1.47	0	1.17	1.17	0	1.96	0	0	10
	1.56	1.56	1.56	0	1.25	1.25	0	2.09	0	0	11
	1.45	1.45	1.45	0	1.16	1.16	0	1.93	0	0	12)

Table 2. Performances (benefit) of the system subject to optimized control laws. Numbers between parenthesis refer to the system performances obtained with the initial estimate of the parameter vector.

no of neurons	system performances	
ν	J (TDC)	J (AEV)
1	29.03 (0.16)	2.98 (0.01)
2	51.95 (0.29)	5.47 (0.03)
3	66.19 (0.46)	7.26 (0.05)
4	74.72 (0.61)	8.42 (0.06)
5	75.53 (0.78)	8.58 (0.07)

law parameters. Feasibility of the control values provided by the optimized control laws was also checked and all the values turned out to satisfy constraint (14b).

5. FINAL REMARKS AND FUTURE RESEARCH

The paper presents an extension of the ERIM to stochastic optimal control problems defined over an infinite horizon. The algorithm that implements such method has been tested on a numerical example with promising results. The application to a real world reservoir network is presently under study.

Further research is also needed for giving formal proof of the convergence properties that were empirically demonstrated by testing the algorithm. In fact, the optimization method is based on a stochastic approximation algorithm where two sources of approximation are present: the one arising from the substitution of the average value $E[Z]$ with a single realization of Z (as in the original version of the ERIM) and the truncation error due to the substitution of an infinite sum with a finite one. As discussed in section 3.4, the choice of the time h at which the truncation is made is also an open issue of the proposed approach.

ACKNOWLEDGEMENTS

Supported by FONDAZIONE CARIPLO TWOLE-2004.

REFERENCES

M. Baglietto, C. Cervellera, M. Sanguineti, and R. Zoppoli. *Topics on System Analysis and Integrated Water Resource Management*, chapter Water Reservoirs Management under Uncertainty by Approximating Networks and Learning from Data. Elsevier, Amsterdam, NL, 2006.

R.E. Bellman. *Dynamic programming*. Princeton University, Press. Princeton, NJ, 1957.

D.P. Bertsekas. *Dynamic Programming and Stochastic Control*. Academic Press, New York, NY, 1976.

H.J. Kushner and G.G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer, New York, 1997.

R. Soncini-Sessa, A. Castelletti, and E. Weber. *Integrated and participatory water resources management. Theory*. Elsevier, Amsterdam, NL, 2007.

S. Yakowitz. Dynamic programming applications in water resources. *Water Resources Research*, 18:673–696, 1982.

R. Zoppoli, M. Sanguineti, and T. Parisini. Approximating networks and extended ritz method for the solution of functional optimization problems. *J. Optim. Theory Appl.*, 112(2):403–440, 2002.