01 Jan 2000

# Infinite Time Optimal Neuro Control for Distributed Parameter Systems

S. N. Balakrishnan
*Missouri University of Science and Technology*, bala@mst.edu

Radhakant Padhi



## Recommended Citation

S. N. Balakrishnan and R. Padhi, "Infinite Time Optimal Neuro Control for Distributed Parameter Systems," *Proceedings of the 2000 American Control Conference, 2000*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2000.
The definitive version is available at https://doi.org/10.1109/ACC.2000.876927

Proceedings of the American Control Conference
Chicago, Illinois • June 2000

# INFINITE TIME OPTIMAL NEURO CONTROL FOR DISTRIBUTED PARAMETER SYSTEMS

***Radhakant Padhi***[1] and ***S.N. Balakrishnan***[2]

Dept. of Mechanical and Aerospace Engineering. and Engineering Mechanics

University of Missouri - Rolla, MO 65409, USA

padhi@umr.edu bala@umr.edu

## Abstract

The conventional dynamic programming methodology for the solution of optimal control, despite having many desirable features, is severely restricted by its computational requirements. However, in recent times, an alternate formulation, known as the *adaptive-critic* synthesis, has given it a new perspective. In this paper, we have attempted to use the philosophy of adaptive-critic design to the optimal control of *distributed parameter systems*. An important contribution of this study is the derivation of the necessary conditions of optimality for distributed parameter systems, described in discrete domain, following the principle of *approximate dynamic programming*. Then the derived necessary conditions of optimality are used to synthesize *infinite time* optimal neuro-controllers in the framework of *adaptive-critic* design. A motivating example that follows clearly shows the potential of the adaptive critic procedure.

## 1. Introduction

The Distributed Parameter Systems (DPS) are processes, which are distributed in space and evolving in time. Unlike the lumped parameter systems, the DPS are described by a set of partial differential equations (PDEs) in the state-space. Examples of such systems include aeroelastic systems, vibration of lightly-damped structures, compliant mechanisms, heat transfer processes etc.

The dynamic programming methodology for the solution of optimal control, despite having many desirable features, is often overwhelmed by its computational requirements. Besides, it is also very involved mathematically. However in recent times, an advanced neuro-control methodology called the *adaptive-critic* design has given a new perspective to it. The advantages include optimal control of the plant maintaining a feedback structure of the controller in real time from any initial state in the domain of interest to the desired final state. Besides, it can handle linear and nonlinear problems directly, retaining the same structure. As an added advantage, the powerful methodology has mathematical and computational simplicity, making it easier for its practical implementation. This neuro-control methodology has been implemented successfully in literature [1,2,3,5]. However, the methodology has mainly been restricted to *lumped parameter systems* so far. In this paper, we derive the necessary conditions of optimality for *distributed parameter systems* in the discrete domain and extend the adaptive-critic methodology for the optimal control of such systems. The focus, however, is the optimal control synthesis of *infinite time* problems. We demonstrate the use of this approach by solving a nonlinear distributed parameter optimal control problem.

## 2. Dynamic Programming of DPS

### 2.1. System Dynamics

We consider a two-dimensional distributed parameter system. *Two dimension* here means two independent variables; one is time and the other is a spatial variable. The system dynamics we consider over here evolves in time and is given by

$$x_{k+1,j} = f_k\left[\left(\bigcup_{j=1}^{M} x_{k,j}\right),\ u_{k,j}\right] \tag{1}$$

where, the subscripts $k$ accounts for evolution with time (time step) and $j$ for the spatial distribution (nodal number). $M$ denotes the final node number in the spatial distribution. The $\bigcup_{j=1}^{M} x_{k,j}$ *notation* denotes that the function $f_k$ may consist of any or all of the variables $x_{k,1}$ , $x_{k,2}$ , ... , $x_{k,M}$ . We expect that the ideas and formulas developed over here can be generalized to systems with higher dimensional distribution in space.

### 2.2. Cost Function

We consider a general cost function of the following form.

$$J = \sum_{k=1}^{N-1}\sum_{j=1}^{M} \Psi_{k,j}\left(x_{k,j},\ u_{k,j}\right) \tag{2}$$

where, $N$ represents the number of discrete time steps. In agreement with the above definition of the cost function, we denote the *cost function from time step* $k$ as

$$J_k = \sum_{\tilde{k}=k}^{N-1}\sum_{j=1}^{M} \Psi_{\tilde{k},j}\left(x_{\tilde{k},j}, u_{\tilde{k},j}\right) \tag{3}$$

Then we can rewrite $J_k$ as

$$J_k = U_k + J_{k+1} \tag{4}$$

[1] Graduate Student, Aerospace Engineering
[2] Professor, Aerospace Engineering (Contact Person)

where,

$$U_k = \sum_{j=1}^{M} \psi_{k,j} \quad and \quad J_{k+1} = \sum_{\tilde{k}=k+1}^{N-1} \sum_{j=1}^{M} \psi_{\tilde{k},j} \tag{5}$$

represent the *utility function* at time step $k$ and the *cost-to-go* from time step $k+1$ to $N$ respectively.

We **define** the *Co-state* as

$$\lambda_{k,j} \equiv \partial J_k / \partial x_{k,j} \tag{6}$$

### 2.3. Optimal Control Equation

For Optimal Control, the necessary condition for optimality is given by $\partial J_k / \partial u_{k,j} = 0$. After some algebra, the optimal control equation is given by

$$\sum_{\tilde{j}=1}^{M} \left( \frac{\partial \Psi_{k,\tilde{j}}}{\partial u_{k,j}} \right) + \sum_{\tilde{j}=1}^{M} \lambda_{k+1,\tilde{j}} \left( \frac{\partial x_{k+1,\tilde{j}}}{\partial u_{k,j}} \right) = 0 \tag{7}$$

### 2.4. Co-state Dynamics

Substituting for $J_k$ from Eq.(3) and expanding the right-hand side partial differentiation, we get

$$\begin{aligned} \lambda_{k,j} = & \sum_{\tilde{j}=1}^{M} \left[ \left( \frac{\partial \Psi_{k,\tilde{j}}}{\partial x_{k,j}} \right) + \lambda_{k+1,\tilde{j}} \left( \frac{\partial x_{k+1,\tilde{j}}}{\partial x_{k,j}} \right) \right] \\ & + \sum_{\tilde{j}=1}^{M} \left[ \sum_{\tilde{j}=1}^{M} \left\{ \left( \frac{\partial \Psi_{k,\tilde{j}}}{\partial u_{k,\tilde{j}}} \right) + \lambda_{k+1,\tilde{j}} \left( \frac{\partial x_{k+1,\tilde{j}}}{\partial u_{k,\tilde{j}}} \right) \right\} \right] \left( \frac{\partial u_{k,\tilde{j}}}{\partial x_{k,j}} \right) \end{aligned} \tag{8}$$

Thus, we have obtained the *state equation, co-state equation* and *optimal control equation.* These equations have to be solved simultaneously to obtain the required optimal control.

### 2.5. Boundary Conditions

We assume that the initial state values at all the node points are *known*, but they can represent ***any profile.*** This implies that *the solution covers all possible initial profiles in the domain of interest.* Here we consider *infinite-time regulator* problems. For this class of problems, the final boundary condition, as derived form the *transversality* condition, is given by $\lambda_{N,j} = 0$ as $N \to \infty$. The other boundary and transversality conditions, considering the Neumann boundary conditions, are given by

$$\begin{aligned} x_{k,0} &= x_{k,1}, \quad x_{k,M} = x_{k,M+1} \\ \lambda_{k,0} &= \lambda_{k,1}, \quad \lambda_{k,M} = \lambda_{k,M+1} \end{aligned} \tag{9}$$

It can be noted that $j = 0$ and $j = M+1$ are not actual node points. However, values at these fictitious node points are needed to obtain the numerical solution.

## 3. Adaptive-Critic Controller Synthesis

The potential of the adaptive-critic approach lies in its architecture. The schematic of critic-controller synthesis procedure is outlined in Figures 1, 2 and 3. We propose a cascade of neural networks, which solves the rigorous optimal control problem, contained in Eq. (1), (7), (8) and (9). More importantly, this set of networks will be able to generate optimal control for ***any arbitrary initial profile***, in the domain for which the networks are trained. The entire synthesis procedure is discussed, in detail, through the following motivating example.

## 4. An Example

This motivating example is taken from Sage and Chaudhuri [4]. Though the authors have considered the problem as a finite time one, we reconsider the problem of an *infinite time controller.* One of the important motivations for choosing this problem is that the problem has already been solved and hence we can at least have a qualitative comparison of the results. The other important reason is that this problem is nonlinear, which demonstrates that the adaptive-critic methodology is not restricted by nonlinear dynamics. Moreover, the problem retains all the essential features of distributed parameter systems.

### 4.1. Problem

The system is described, in continuous time, by

$$\frac{\partial x(t,y)}{\partial t} = -x^3(t,y) + \frac{\partial x^2(t,y)}{\partial y^2} + u(t,y) \tag{10}$$

The known initial and boundary conditions are as follows.

$x(0,y) \equiv$ An envelope of initial profiles within the domain of interest

$$\left. \frac{\partial x(t,y)}{\partial y} \right|_{y=y_0} = 0, \qquad \left. \frac{\partial x(t,y)}{\partial y} \right|_{y=y_f} = 0 \tag{11}$$

The objective is to find the *optimal control* $u(t,y)$, which minimizes the *quadratic cost function*

$$J = \frac{1}{2} \int_{t_0}^{\infty} \int_{y_0}^{y_f} \left[ Qx^2(t,y) + Ru^2(t,y) \right] dy\,dt \tag{12}$$

where, $x(t,y)$ and $u(t,y)$ are state and control variables at time $t$ and spatial co-ordinate $y$, $Q$ is the *weighting factor* on state, $R$ is the *weighting factor* on control, $t_0$ and $t_f \to \infty$ are initial and final times, $y_0$ and $y_f$ are initial and final points on the spatial co-ordinate axis.

### 4.2. Discrete Formulation

The discretized cost-function, to be minimized, is given by

$$J = \frac{1}{2} \left[ \sum_{k=1}^{\infty} \sum_{j=1}^{M} \left( Q_D \, x_{k,j}^2 + R_D \, u_{k,j}^2 \right) \right] \tag{13}$$

where, $Q_D$ and $R_D$ are the weighting factors on state and control respectively, *in the discrete domain.* For this particular problem,

$$\Psi_{k,j} = \frac{1}{2}\left[ Q_D\, x_{k,j}^2 + R_D\, u_{k,j}^2 \right] \quad (14)$$

Then, by using Eq.(1), (7) and (8), we arrive at the following set of equations as the necessary conditions for optimality, which are the *state, co-state* and *optimal control* equations respectively.

$$x_{k+1,j} = x_{k,j} + \left[ (1/\Delta y^2)\left(x_{k,j+1} - 2x_{k,j} + x_{k,j-1}\right) - x_{k,j}^3 + u_{k,j} \right] \Delta t \quad (15a)$$

$$\lambda_{k,j} = \lambda_{k+1,j} + \begin{bmatrix} Q_D\, x_{k,j} + (1/\Delta y^2)(\lambda_{k+1,j+1} - 2\lambda_{k+1,j} + \lambda_{k+1,j-1}) \\ -3\lambda_{k+1,j} x_{k,j}^2 \end{bmatrix} \Delta t \quad (15b)$$

$$u_{k,j}^* = -R_D^{-1}\, \lambda_{k+1,j} \quad (15c)$$

Together with the necessary conditions of optimality, we have to satisfy the following initial, transversality and boundary conditions.

$x_{0,j}$ = can be *any point* in the domain of interest

$$\lambda_{N,j} = 0, \quad as\ N \to \infty$$

$$x_{k,0} = x_{k,1}, \qquad x_{k,M} = x_{k,M+1} \quad (16)$$

$$\lambda_{k,0} = \lambda_{k,1}, \qquad \lambda_{k,M} = \lambda_{k,M+1}$$

### 4.3. Adaptive-Critic Controller Synthesis

The architecture of neural networks in the optimal control problem of this motivating example remains the same as that of the general architectures, outlined in Figures 1, 2 and 3. The synthesis procedure of the networks is discussed, in detail, in this section.

From the transversality condition [Eq.(16)], we have $\lambda_{N,j} = 0, \quad N \to \infty$. Thus, from Eq.(7), we have $\partial \Psi_{N-1,j} / \partial u_{N-1,j} = 0$. In a regulator case, this leads to $u_{N-1,j} = 0$. However, *with* $\lambda_{N,j} = 0$, from Eq.(15b) we obtain $\lambda_{N-1,j} = Q_D\, x_{N-1,j}$. With $u_{N-1,j} = 0$, we also notice that we can solve for $x_{N-1,j}$ *explicitly*, from Eq.(15a), knowing $x_{N,j}$ and using the boundary condition (Eq. 16). Hence, we synthesize a set of $M$ critic networks, for $k = N-1$, with input $x_{N-1,j}$ and output $\lambda_{N-1,j}$ as per the following steps.

Assume a set of random values for $x_{N,1}, \ \ldots, x_{N,M}$ with *very small* magnitude, compared to the initial magnitude. Solve the system of $M$ *coupled equations* simultaneously from the state Eq.(15a), with $u_{N-1} = 0$ to get $x_{N-1,1}, \ \ldots, x_{N-1,M}$. Use of boundary condition [Eq.(16)] makes the system into $M$ coupled equation with $M$ unknowns and hence makes it possible to solve. Since the problem is *nonlinear* and hence, the solution is not unique, we find the solution for which $x_{N-1,j} \to 0$, for all $j = 1, \ldots, M$. Obtain $\lambda_{N-1,1}, \ \ldots, \lambda_{N-1,M}$ from $\lambda_{N-1,j} = Q_D\, x_{N-1,j}$. Initialize a separate neural network for node $j$ and train it with **input** $x_{N-1,j}$ and **output** $\lambda_{N-1,j}$. One can notice that this synthesis procedure *does not* require the use of action networks.

Then we focus on action network synthesis. However, since it is a dynamic process, to begin with, it is desirable to have 'good' initial weights of the neural networks, for faster final convergence. For this reason, we pre-train the randomly initialized networks first. The essential idea of the *pre-training* process is to generate $x_{k,j}$ close to $x_{k+1,j}$ and then to use $x_{k,j}$ as if it were $x_{k+1,j}$. The detailed steps can be outlined as follows.

Assume random $x_{k,j}$, close to $x_{k+1,j}$. Assume $x_{k,j}$ as $x_{k+1,j}$ and input it to the *trained* set of critic networks at $(k+1)^{\text{th}}$ time step to get $\lambda_{k+1,j}$. Get the optimal control $u_{k,j}^*$ from Eq.(15c). Initialize a separate neural network for each node point and train the networks at $k^{\text{th}}$ time step with **input** $x_{k,j-1}, x_{k,j}, x_{k,j+1}$ and **output** $u_{k,j}^*$ for all the networks related to the *internal node points*. For those intended for the *boundary node points*, we consider either $x_{k,0}, x_{k,1}$ or $x_{k,M-1}, x_{k,M}$ as the **input.**

Once the pre-training is over, actual training process is carried out in the following steps (see Figure 2).

Assume random $x_{k,j}$, within the relevant range, and input it to the pre-trained action networks, to get $u_{k,j}$. Use *state equation* (15a) and the boundary condition [Eq.(16)] to get $x_{k+1,j}$ uniquely.

Input $x_{k+1,j}$ to the *trained* set of critic networks to get $\lambda_{k+1,j}$. Get the optimal control $u_{k,j}^*$ from Eq.(15c). Train the networks at $k^{\text{th}}$ time step with **input** $x_{k,j-1}, x_{k,j}, x_{k,j+1}$ and **output** $u_{k,j}^*$ for all the networks related to the *internal node points*. For those intended for the *boundary node points*, we consider either $x_{k,0}, x_{k,1}$ or $x_{k,M-1}, x_{k,M}$ as the **input**.

Once this process is over and proper convergence of all the networks is realized, we go back to the critic training. However, from this time onwards action networks are also used in the synthesis process of critic networks and *vice versa*. Each time we revert to critic training, we also increase the boundary of the state values $x_{k,j}$ (typically by 5%) and train the networks for random initial profiles within the new increased boundary. The detail process of critic training can be outlined as follows (see Figure 3).

Assume random $x_{k,j}$. Get $u_{k,j}$ from the trained action networks. Get $x_{k+1,j}$ from the *state equation* (15a). Input $x_{k+1,j}$ to the trained set of critic networks at $(k+1)^{\text{th}}$ time step, to get $\lambda_{k+1,j}$. Now, with the availability of $x_{k,j}$ and $\lambda_{k+1,j}$, calculate $\lambda_{k,j}$ from the *co-state* equation (15b). Use of transversality condition [Eq.(16)] makes it possible to calculate $\lambda_{k,j}$ uniquely. Train the set of critic networks with **input** $x_{k,j-1}$, $x_{k,j}$, $x_{k,j+1}$ and **output** $\lambda_{k,j}$ for all the networks related to the *internal node points*. For those intended for the *boundary node points*, we consider either $x_{k,0}$, $x_{k,1}$ or $x_{k,M-1}$, $x_{k,M}$ as **input**.

Once this process of critic synthesis is over, we revert back to action synthesis again. The process outlined is repeated iteratively till we sufficiently increase the boundary for the input state values in both the action and critic networks to a level within which the initial profiles are supposed to lie. It can be noted that when the final boundary is reached, the alternate critic and action network training process is continued till no noticeable change in the output is observed in the outputs in the successive training. Then the networks are supposed to converge to give the true optimal relationships between their input and output.

## 5. Numerical Results

In our numerical experiments, we set the numerical values as $Q=1$, $R=1$, $t_0=0$, $y_0=0$, $y_f=4$ (same as Ref..[4]). For discretization, we used $\Delta t=0.02$, $\Delta y=1$. Consequently, we realize *five node points*, which are numbered as 1, ..., 5 in Figure 4 through Figure 7. We have assumed that the initial profiles lie within $\pm 0.25$ and hence, stopped the synthesis process when we finished training at this level, for the states at the node points, upon successive increase of the magnitude boundary for the initial profiles.

In Figures 6 and 7 the state trajectories and the associated optimal control histories of the system for an initial arbritary random profile are shown. It should be noted that the simulation time has been arbitrarily fixed at 8 sec. though it can be continued till any time. If done so, the states and control histories all remain very close to zero values. As a comment, we also observe from Eq.(15c), that $\lambda_N \to 0$ as $N \to \infty$, since $u_N \to 0$ as $N \to \infty$. This way the synthesis process is also seen to satisfy the necessary *transversality* condition for optimality given in Eq.(16). We have also plotted the state trajectories and optimal control histories for all the node points for a sinusoidal initial profile. These also give a clear picture so as how the states and control are driven towards zero together, at all the nodes.

It can be noted that after synthesizing the series of action networks, the controller can be implemented ***on-line,*** since the computations involve only *using* the trained neural networks. Moreover, since the *current states* essentially determine the control required, the control law takes a *feedback* structure. This is an added advantage.

## 6. Conclusion

The necessary conditions of optimality for distributed parameter systems, described in discrete domain, have been derived following the principle of approximate dynamic programming. The *adaptive-critic* optimal neuro-control is used to obtain the optimal control of distributed parameter systems. The methodology retains all the beneficial features of dynamic programming. It is possible to have optimal control solutions for a countably *infinite number of initial profiles*, and *in real-time.*


### Acknowledgement:

This research was supported by NSF Grant ECS-9976588.


## References


1. White D. A. and Sofge D., *Handbook of Intelligent Control*, Van Nostrand Reinhold, 1992.
2. Balakrishnan S. N. and Biega V., *Adaptive-Critic Based Neural Networks for Aircraft Optimal Control*, Journal of Guidance, Control and Dynamics, Vol. 19, No. 4, July-Aug. 1996, pp. 893-898.
3. Han D. and Balakrishnan S. N., *Adaptive-Critic Based Neural Networks for Agile Missile Control*, AIAA-98-4495.
4. Sage A. P. and Chaudhuri S. P., *Gradient and Quasi-linearization Computational Techniques for Distributed Parameter Systems*, International Journal of Control, Vol. 6, No. 1, 1967, pp. 81-98.
5. Werbos P., *Neurocontrol and Supervised Learning: An Overview and Evaluation*, Handbook of Intelligent Control, Van Nostrand Reinhold, 1992.


## Figures

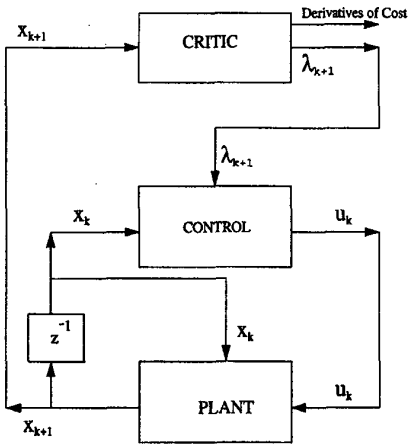


Figure 1: Successive training of Critic and Action Networks

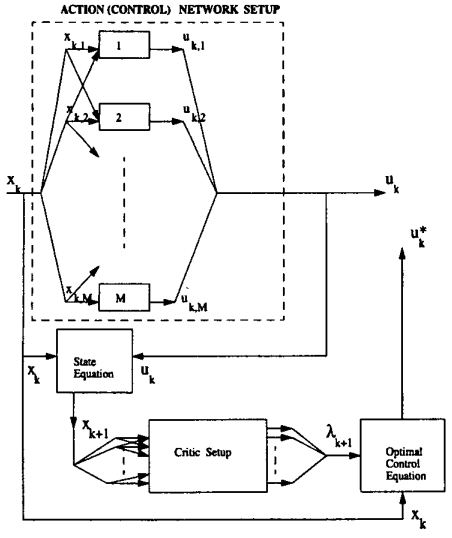


Figure 2: Schematic of Action Synthesis Procedure

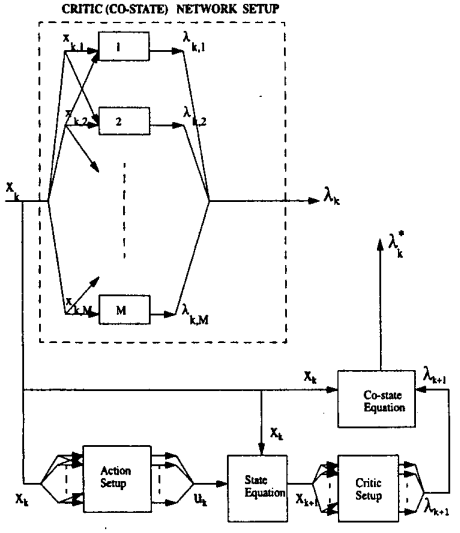


Figure 3: Schematic of Critic Synthesis Procedure

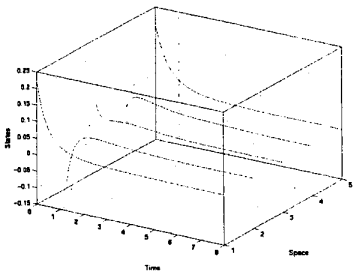

Figure 4: State Trajectories from a Random Initial Profile

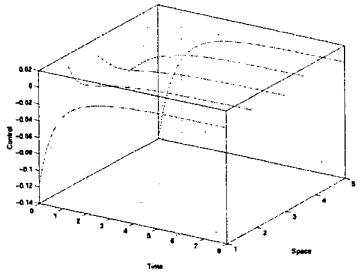

Figure 5: Control Histories for the Random Initial Profile

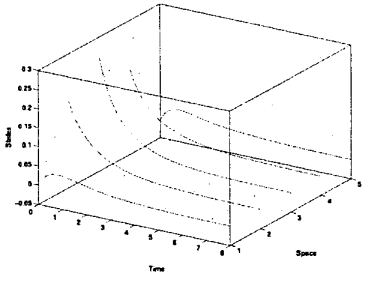

Figure 6: State Trajectories from a Sinusoidal Initial Profile

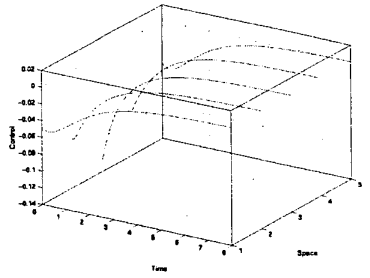

Figure 7: Control Histories for the Sinusoidal Initial Profile