6-4-2003

Design of Predictive Controllers by Dynamic Programming and Neural Networks

Chen-Wen Yen
National Sun Yat-Sen University

Mark L. Nagurka
Marquette University, mark.nagurka@marquette.edu

Design of Predictive Controllers by Dynamic Programming and Neural Networks

Chen-Wen Yen
National Sun-Yat Sen University

M.L. Nagurka
Department of Mechanical Engineering, Marquette University, Milwaukee, WI

SECTION 1. INTRODUCTION

Predictive control methods have many exceptional features, including their broad applicability to a wide variety of systems and their ease of tuning [1]. However, only a few predictive controller design methods are applicable to nonlinear systems [2]–[3][4][5][6][7][8]. These approaches suffer from one or more of the following limitations: they may not be able to handle constraints, they may not guarantee a global optimal solution, and they may not be able to specify a closed-loop control law.

SECTION 2. PROBLEM STATEMENT

With y as the output variable, u as the control variable, ω as the reference trajectory, $k = 0$ as the current sampling instant, H_P as the perdition horizon, H_m as the minimum horizon and H_c as the control horizon, the goal of a predictive control problem is to find a control law for the following system,

$$y(k+1) = f(y(k), \cdots, y(k-n), l1(k), \cdots, u(k-m)) \quad (1)$$

that minimizes the criterion function,

$$J = \sum_{i=H_m}^{H_p} [\hat{y}(i) - \omega(i)]^2 \qquad (2)$$

while satisfying the constraints,

$$\begin{gathered} y_{min} \le y(i) \le y_{max} 1 \le i \le H_p \\ u_{min} \le u(i) \le u_{max} 1 \le i \le H_p - 1 \\ \Delta u_{min} \le \Delta u(i) \le \Delta u_{max} 1 \le i \le H_p - 1 \\ u(i) = u(H_c - 1) i > H_c - 1 \end{gathered} \qquad (3)(4)(5)(6)$$

where $\Delta u(i) = u(\imath) - u(i-1)$ and where the predictive output in Eq. (2) is

$$\hat{y}(k+i) = f(z \quad (k+i-1), \cdots, z(k+i-n), \\ u(k+i-1), u(k+i-m)) \qquad (7)$$

View Source (following [1]) with

$$\begin{gathered} z(j) = y(j) if j \le 0 \\ z(j) = \hat{y}(j) if j > 0. \end{gathered} \qquad (8)$$

Eqs. (3) and (4) represent constraints on the operational regions of the output and control variables, respectively. Eq. (5) limits the rate of change of the controller output and Eq. (6) constrains the controller output to be a constant when the control process moves H_c sampling periods into the future.

SECTION 3. A DP-BASED SOLUTION APPROACH

The problem defined in the previous section can be viewed as an optimal control problem and can be solved via dynamic programming (DP). DP offers several advantages: *(i)* it can handle constraints easily, *(ii)* it gives the global minimum solution, and *(iii)* it provides closed-loop solutions. This study relies on DP to develop a predictive control law.

To simplify the development process, the system model of Eq. (1) is rewritten as

$$y(k+1) = h(x(k), \Delta u(k)) \qquad (9)$$

where

$$\mathbf{x}(k) = \quad [v(k), y(k-1), \cdots, y(k-n), \\ u(k-1), u(k-2), \cdots, u(k-m)]^T. \qquad (10)$$

Also, the cost associated with the *i*th stage of the control process is expressed as

$$g(\boldsymbol{x}(i-1),\Delta u(i-1)) = [y(i)-\omega(i)]^2 = [h(\boldsymbol{x}(i-1),\Delta u(i-1))-\omega(i)]^2 \quad (11)$$

By applying the principle of optimality [9], it can be shown that the general recurrence relation for the optimal control law is

$$J^*_{H_c-k,H_P}(\mathbf{x}(H_c-k)) = \min_{\Delta u(H_c-k)}[g(\mathbf{x}(H_c-k),\Delta u(H_c-k)) + J^*_{H_c-k+1,H_P}(\mathbf{x}(H_c-k+1))] \quad (12)$$

where $J_{m,M}(x(m))$ denotes the portion of the criterion function associated with the time interval from the mth to the Mth sampling instant for the given $x(m)$ and that J^* represents the optimal value of J. In addition, in the absence of Δu constraints, the recurrence relation becomes

$$J^*_{H_c-k,H_P}(\mathbf{x}(H_c-k)) = \min_{\Delta u(H_c-k)}[q(\mathbf{x}(H_c-k),\Delta u(H_c-k)) + J^*_{H_c-k+1,H_P}(\mathbf{x}(H_c-k+1))] \quad (13)$$

where

$$q(\mathbf{x}(i-1),u(i-1)) = [y(i)-\omega(i)]^2 = [f(\mathbf{x}(i-1),u(i-1))-\omega(i)]^2 \quad (14)$$

To implement the control policy in a closed-loop configuration, the DP generated solutions should be stored in *high-speed* memory so that appropriate control signals can be retrieved online by means of a suitable table look-up method. However, this constitutes a formidable storage burden since the memory requirement increases exponentially with the order of the system. This "curse of dimensionality" problem is resolved in the next section.

SECTION 4. THE NEURAL CONTROL METHOD

This section introduces a neural network based approach to replace the conventional *table* look-up method. For simplicity, the approach as described is applicable only to problems with Δu constraints. (The approach can be modified to handle problems without Δu constraints.)

To learn the DP generated control law, a regression neural network (RNN) method can be proposed with the following steps:

1. Use a trajectory planner to generate the desired output response y_d.
2. Set the reference trajectory $\omega = y_d$ and use the previously described DP-based solution method to compute the predictive control law.
3. With data provided by the DP-based control law, train a RNN to simulate the mapping from $x(k)$ to $\Delta u(k)$

This direct method, however, has two drawbacks. First, the neural network training process must be repeated when the desired output response has been changed. This restricts the practicality of the control method.

Second, due to the existence of constraints, the control law may not be a continuous or continuously differentiable function of $x(k)$. As a result, accurate neural network training may be difficult.

To resolve the first problem, this study uses the following first-order reference trajectory to approximate the desired output response y_d,

$$\begin{aligned}\omega(k+1) \quad &= (1-\alpha)y_d(k+H_P) + \\ &\alpha\omega(k+i-1) 1 \le i \le H_p 0 \le \alpha < 1\end{aligned} \quad (15)$$

with $\omega(k) = y(k)$. By generating the reference trajectory ω in this manner, the approach characterizes any desired output response y_d by $y(k)$ and $y_d\{k+H_p\}$ when ω is given. Treating $y(k)$ and $y_d\{k+H_p\}$ as trajectory parameters, this study uses the following procedure to create training samples so that neural networks can be trained to generate a trajectory dependen[t] predictive control law:

1. Quantize the admissible $y(k)$ and $y_d(k+H_p)$ values into a fi.nite number of levels.
2. For every quantized set of $[y(k)y_d(k+H_p)]$, use the proposed DP solution method to find the predictive control law.
3. With $x(k)$ and $y_d(k+H_p)$ as inputs and $\Delta u(k)$ as output, use results obtained from the previous step to train neural networks. Note that $y(k)$ is an element of *x(k)*.

Since the trajectory parameters $y(k)$ and $y_d(k+H_p)$ are part of the neural network inputs, they can be varied to account for desired output response changes. For convenience, the input vector is denoted as z where $z^{\mathrm{T}}(k) = [x^{\mathrm{T}}(k)y_d(k+H_p)]$.

To resolve the second problem, a divide-and-conquer strategy is proposed to divide the potentially *discontinuous* mapping learning problem into several continuous mapping learning problems. In particular, before the application of the RNN, a classification neural network (CNN) is first used to learn to classify the predictive control policy into the following five patterns

$$u(k) = \begin{cases} u_{max} \\ u_{min} \\ u(k-1) + \Delta u_{max} \\ u(k-1) + \Delta u_{min} \\ otherwise \end{cases} \quad (16)$$

Hereafter, the first four patterns are referred to as the $u_{max}u_{min}, \Delta u_{max}$ and Δu_{min} control patterns, respectively. The last control pattern is called the nonsaturated control pattern. Note that the control signals associated with u_{max} and u_{min} control patterns are apparently known and the values of $u(k)$ associated with the Δu_{max} and Δu_{min} patterns can also readily be determined since $u(k-1)$ is one of the CNN inputs. Consequently, with the assistance of the CNN, the RNN only needs to learn to simulate the nonsaturated control pattern. The possibility of using a RNN to learn a discontinuous mapping is thus avoided.

The block diagram of Fig. 1 depicts the structure of the proposed two-stage neural control method. In the first stage, a CNN module is used to classify the $z(k)$ signal into one of five predefined categories. The control signals associated with the first four control patterns are determined with the CNN whereas the RNN is used to generate nonsaturated control commands corresponding to a continuous mapping region. (Although not shown in Fig. 1, the RNN has the same inputs as the CNN.)

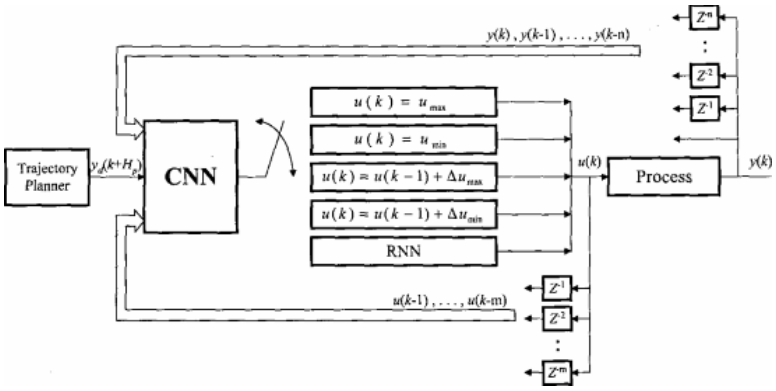

Fig. 1. The block diagram of the proposed neural control method.

SECTION 5. EXPERIMENTAL STUDIES

Two predictive control problems for a DC motor servo system are posed to test the effectiveness of the method. The experimental setup uses a PC to perform the I/O actions. The axes of an X-Y table are actuated by two linear voltage amplifier driven DC motors. The model of each DC motor servo system is of the following form,

$$y(k+1) = ay(k) + bu(k) + c\text{sgn}(y(k)) + d \qquad (17)$$

where y is the angular velocity of the DC motor, u is the armature voltage and sgn is the signum function used to account for the effect of friction. The parameters of the model were determined by a standard least-squares identification method. The mean and variance of the trajectory tracking error, $e = y_d - y$, were used to characterize the tracking efficacy.

In applying the neural control method, the one-hidden-layer optimal interpolative (OI) neural network and training method proposed by Sin and Defigueiredo [11] was used to construct the CNN. The implementation of the RNN is based on the one-hidden-layer radial basis function (RBF) neural network and the training algorithm developed by Chen and Billings [12].

The tested control methods were used to track two different trajectories. In the first case, the desired output response was $y_a(k) = 700\text{sin}(2\pi k/500)\text{rpm}$. The desired output response of the second case was a 5 second period (50% duty *cycle)* square-wave function whose velocity changed between 700 and −700 rpm. The length of both control processes was 20 seconds. For the square-wave tracking problem, the transition period data were excluded in computing the means and variances of the trajectory tracking errors so that the regulator behavior *of the* tested control methods could be examined more accurately. For reliability, the experiment was repeated five times.

Based on restrictions imposed by the hardware, the following constraints were included,

$$\begin{gathered} -1000\text{rpm} \le y(i) \le 1000\text{rpm} 1 \le i \le H_p \\ -33\text{V} \le u(i) \le 33\text{V} 0 \le i \le H_p - 1 \end{gathered} \qquad (18)(19)$$

corresponding to a maximum speed limit of 1000 rpm and an armature voltage limit of 33 V. To simulate a deadbeat controller, the prediction horizon H_p, the minimum horizon H_m, and the control horizon H_c were chosen as 10, 1, and 3, respectively [1].

In applying the proposed DP based solution approach, the admissible y and u regions were both quantized into 101 grid points while the admissible region of $y_d(H_p)$ was quantized into 41 grid points. The reference trajectory parameter ω was chosen as 0.1. With these settings, DP found 3892 sets of admissible solutions containing 1291 u_{max} control pattern data, 1043 u_{min} control pattern data and 1558 nonsaturated control pattern data. The number of floating point variables that needed to be stored was 11676. With these data, the OI network based CNN achieved a 99% classification accuracy with 12 hidden neurons. For the RNN part, a 50-hidden-node RBF network was employed. Overall, the proposed neural control method required 220 floating point number memory space, representing a 98% storage space reduction.

The tracking performance of the proposed approach was compared to that of the conventional table look-up control method. The two methods were used to track the two desired output responses described in the beginning of this section. The experimental results, summarized in Tables 1 and 2, indicate that the tracking performance of the proposed approach – with its significant saving in the storage requirement – is not inferior to that of the table look-up method. To explain this phenomenon, recall that the DP-based method generates the predictive control law only at a finite number of grid points. In the remaining part of the ***x*** space, the control policy is determined by interpolation. The table look-up method is thus subject to interpolation errors. The comparable tracking results demonstrate that the generalization error of the neural networks is not necessarily larger than the interpolation error associated with the table look-up method. The results show that the method can reduce the storage space requirement by a factor of 50.

Table 1: Summary of trajectory tracking errors for example 1: Sinusoidal output response case.

Experiment Number	Table Look up method		Neutral Controller	
	Mean (rpm)	Variance (rpm^2)	Mean (rpm)	Variance (rpm^2)
1	1.58	41.76	1.91	31.99
2	1.53	44.12	1.88	31.65
3	1.54	42.67	1.98	31.70
4	1.70	44.84	2.27	30.33
5	1.46	43.69	2.24	30.06
Average	1.56	43.42	2.06	31.15

Table 2: Summary of trajectory tracking errors for example 1: Square wave output response case.

Experiment Number	Table Look up method		Neutral Controller	
	Mean (rpm)	Variance (rpm^2)	Mean (rpm)	Variance (rpm^2)
1	2.62	24.26	-0.02	15.80
2	2.62	18.76	-0.01	19.40
3	2.46	21.83	-0.13	21.31
4	2.54	20.75	0.01	19.23
5	2.62	20.38	-0.08	15.11
Average	2.57	21.20	-0.03	18.17

SECTION 6. CONCLUSION

This paper proposes a dynamic programming (DP) and neural network based predictive controller design method for nonlinear systems. The DP based solution method has several exceptional features. First, it deals with constraints easily. Second, it finds the global optimal solution. Third, it produces a closed-loop control law. To overcome the potential "curse of dimensionality" problem associated with the DP generated control law, neural networks are used to reduce the storage requirement. The proposed approach can achieve significant storage space saving without sacrificing tracking efficacy. In addition to the significant saving in memory space,

experimental results demonstrate that the control method provides tracking performance comparable to that of a conventional approach.

ACKNOWLEDGEMENT

Prof. C-W. Yen acknowledges the support of the National Science Council of the Republic of China under Grant Number NSC 84–2212-E-110–033. Prof. M.L. Nagurka is grateful for a Fulbright Scholarship for the 2001–2002 academic year allowing him to pursue this research at the Weizmann Institute of Science.

REFERENCES

1. Soeterboek, R., 1992, Predictive Control: A Unified Approach, Prentice-Hall, Englewood Cliffs, NJ.
2. Hernádez, E., and Arkun, Y., 1993, "Control of Nonlinear Systems Using Polynomial ARMA Model," AIChE Journal, 39, 446-460.
3. Patwardhan, S. C., and Madhavan, K. P., 1993, "Nonlinear Model Predictive Control Using Second-Order Model Approximation," Ind. Eng. Chem. Res., 32, pp. 334-344.
4. Bittanti, S., and Piroddi, L., 1994, "GMV Technique for Nonlinear Control with Neural Networks," IEE Proc.-Control Theory Appl., 141, pp. 57-69.
5. Chen, Q., and Weigand, W. A., 1994, "Dynamic Optimization of Nonlinear Processes by Combining Neural Net Model with UDMC," AICheE Journal, 40, pp. 1488-1497.
6. Sistu, P. B., Gopinath, R. S., and Bequette, B. W., 1994. "Computational Issues in Nonlinear Predictive Control," Computers Chem. Eng., 17, pp. 361-366.
7. Doyle, F. J., Ogunnaike, B. A., and Pearson, R. K., 1995. "Nonlinear Model-Based Control Using Second-Order Volterra Models," Automatica, 31, pp. 697-714.
8. Lu, P., 1995, "Optimal Predictive Control of Continuous Nonlinear Systems," Int. J. Control, 62, pp. 633-649.
9. Kirk, D. E., 1970, Optimal Control Theory: An Introduction, Prentice-Hall, Englewood Cliffs, NJ
10. Luus, R., 1990, "Optimal Control by Dynamic Programming Using Systematic Reduction in Grid Size," Int. J. Control, 51, pp. 995-1013.
11. Sin, S.-K., and Defigueiredo, R. J. P., 1993, "Efficient Leaning Procedures for Optimal Interpolative Nets," Neural Networks, 6, pp. 99-113.
12. Chen, S., and Billings, S. A., 1992, "Neural Networks for Nonlinear Dynamics System Modeling and Identification," Int. J. of Control, 56, pp. 319-346.

IEEE Keywords

Dynamic programming , Neural networks , Control systems , Optimal control , Sampling methods , Design methodology , Nonlinear control systems , Nonlinear systems , Gold , Process control