Learning Control by Numerical Optimization Methods[1]

R. W. Longman[2], Homayoon S. M. Beigi[3], and C. James Li[4]
Columbia University
New York, New York 10027

ABSTRACT

When control systems are given the same task to perform repeatedly, they will usually repeat the same errors in executing the command, except for certain random noise effects. The field of learning control has developed to produce controllers that can improve their performance at a given task with each repetition of the task. To date, learning control algorithms have concentrated on use of analogues of integral control to eliminate errors, and recently on conversions of adaptive control to the learning control problem. This paper studies an alternative of using numerical optimization techniques to determine ways to improve controller performance from one repetition to the next. Quasi-Newton methods, the direction set method, and certain derivative free algorithms are studied. Results seem to indicate that methods involving system identification in the repetition domain will converge to zero tracking error in fewer repetitions than methods aimed at direct minimization of the Euclidean norm of the error.

INTRODUCTION

A large percentage of the practical applications of control theory involve systems which are repeatedly asked to perform the same task. Examples include tracking problems for robots on assembly lines, as well as a large number of manufacturing applications. Standard controller design methods produce systems that do an imperfect job of executing a command -- and when the command is repeated the system repeats these same errors everytime. It is perhaps a bit primitive to persist in repeating the same errors, and in the last few years a theory of learning control has been developed in the literature, generating controllers that can learn from previous experience at performing a specific task [1-7]. The fact that a specific task is involved repeatedly, distinguishes the learning control problem from the adaptive control problem.

To date the learning control literature has been based on either analogues of integral control in the repetition domain [1-4], or more recently adaptive control ideas have been converted to apply to the learning control problem in [5-7]. This paper is devoted to a preliminary investigation of a third option -- the use of numerical optimization techniques as a method of accomplishing learning in repetitive tasks.

The approach bears a certain philosophical similarity to a few optimal control investigations that use the real world rather than a computer model to evaluate each iteration in the attempt to converge on the optimal control. Such an approach can avoid the sometimes lengthy process of obtaining a good model before the optimization problem can start. Reference [3] develops such an idea for minimizing the integral of the absolute value of the error in continuous systems.

Here, we consider the learning control problem for discrete systems, and formulate an optimization function which is quadratic in the error. We study numerical optimization methods to determine how to adjust the control history from one repetition to the next in order to decrease this quadratic function of the error as the repetitions of the task proceed.

PROBLEM FORMULATION

We consider a general discrete time linear time-varying or time-invariant system,

$$x_j(k+1) = A(k)\, x_j(k) + B(k)\, u_j(k) + w_j(k) \qquad (1)$$
$$y_j(k) = C(k)\, x_j(k)$$
$$k=0,1,2,\ldots,p-1$$
$$j=0,1,2,\ldots$$

[1] Research partially supported by NASA Grant NAG 1-649.
[2] Professor of Mechanical Engineering.
[3] Graduate Student.
[4] Assistant Professor of Mechanical Engineering.

where the state is n-dimensional, the control is m-dimensional, the output is q-dimensional ($q \geq m$), k is the time step in the p-step repetitive operation, and j is the repetition number. For simplicity, the dimension n of the system is assumed known, but generalization to knowing an upper bound on n is easily considered. Also, A, B, C, and w_j are assumed unknown -- otherwise one could determine in advance what control to use to minimize tracking error, and there would be no need for learning control. In the learning control problem (as contrasted with the repetitive control problem in [2]) the system is assumed to always start from the same initial state in each repetition of the task. Matrix A includes any state or output feedback control present in the system and the symbol u_j is reserved for the signal added to the control for learning purposes. A time varying model is considered because many applications such as in robotics involve nonlinear dynamic systems which when linearized produce linear models with coefficients that vary with time step, and vary in the same manner each repetition. In such repetitive operations it is often the case that there will be disturbances $w_j(k)$ that repeat with each repetition of the task, and the learning can be made to also correct for this source of errors in a natural way. One of the purposes of the feedback control is to handle any nonrepetitive disturbances, and these will be ignored for purposes of designing the learning control (some analysis of these random noise effects in learning control can be found in [4]).

The solution to (1) for the p-steps of the repetitive process can be written as

$$x_j(k+1) = \Big(\prod_{i=0}^{k} A(t)\Big) x_j(0) + \sum_{i=0}^{k} \Big(\Big(\prod_{r=i}^{k} A(r)\Big)(B(t)\, u_j(t) + w_j(t))\Big) \qquad (2)$$

where the product symbol is taken to give the identity matrix if the lower limit is larger than the upper limit. Defining a difference operator $\delta_j z(k) = z_j(k) - z_{j-1}(k)$ for any variable z, and using the fact that $x_j(0)$ and $w_j(k)$ are repetitive, one can write

$$y_{j+1} = y_j + P\, \delta_{j+1} u \qquad (3)$$

where

$$y = [y^T(1)\ \ y^T(2)\ \ \ldots\ \ y^T(p)]^T$$

$$u = [u^T(0)\ \ u^T(1)\ \ \ldots\ \ u^T(p-1)]^T$$

$$P = \begin{bmatrix} C(1)B(0) & 0 & \cdots & 0 \\ C(2)A(1)B(0) & C(2)B(1) & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot \\ C(p)A(P-1)A(P-2)\ldots A(1)B(0) & \cdots & \cdots & C(p)\,B(p-1) \end{bmatrix}$$

The objective is to find a change in the control at each step j that will as j progresses minimize the quadratic error function

$$f(\delta_{j+1} u) = (y_{j+1} - y_D)^T (y_{j+1} - y_D) \qquad (4)$$

where y_D is the desired output history for the p-step process. In the previous works on learning control in discrete systems [4-7], as well as in the extensions of these papers for journal publication, attention had to be directed to the problem of insuring that the special desired trajectory was in fact a feasible trajectory for the system to perform. For example, one method involved assuming knowledge of the system dimension n, and assuming the system is controllable, and then specifying the desired measured output variable history every n steps, which guaranteed to be a feasible specification by modern control theory. In the present approach these conditions can be relaxed if one is satisfied with minimizing (4) rather than insisting that (4) be driven to zero.

For simplicity of notation, denote $\delta_{j+1} u$ by v_j. Then use of (3) in (4) produces

$$\begin{aligned} f(v_j) &= (y_j + P\, v_j - y_D)^T (y_j + P\, v_j - y_D) \\ &= v_j^T (P^T P)\, v_j + 2\, v_j^T [P^T (y_j - y_D)] + (y_j - y_D)^T (y_j - y_D) \end{aligned} \qquad (5)$$

Note that the gradient of this objective function and the Hessian matrix of second partials are

$$\frac{df}{dv_j} = 2\, P^T P\, v_j + 2\, P^T (y_j - y_D) \qquad (6)$$

$$\left[\frac{d^2f}{dv_j^2}\right] = 2\,P^TP \qquad (7)$$

It is now possible to characterize the nonlinear optimization problem represented by our learning control objective. We wish to

1. minimize a function which is known to be quadratic in the control change variable v_j,
2. but the first derivative of this function is not directly available, because we do not know the system matrices A, B, C, and hence do not know P,
3. and the second derivative is similarly unavailable.

Having characterized the problem, we now present a preliminary assessment of the possible approaches to the learning control problem.

QUASI-NEWTON METHODS

Over the last couple of decades various finely tuned quasi-newton based methods have been generated for the minimization of unconstrained nonlinear functions. Among these methods are the rank 1 and rank 2 methods of DFP and BFGS, and other members of the Broyden family of methods [8]. An important characteristic of these methods is the use of the iterative process to approximate the Hessian matrix so that no explicit expression for the second derivative is needed. If we could evaluate the first derivative df/dv_j, then the BFGS method would converge to the optimal v in approximately mp repetitions. One might note that if one actually knows the df/dv_j of (6), then one could equate it to zero and solve for v and obtain the optimal v in one step -- something which could be classified as a Newton method in numerical optimization.

Since we do not know df/dv_j, one can consider use of quasi-newton methods with a finite difference approximation of the gradient. In order to obtain these differences for all of the mp elements of the gradient vector it would require approximately mp repetitions of the task to make one evaluation of the gradient vector. The total number of repetitions required for convergence would exceed $(mp)^2$. This makes such methods a poor choice in learning control. Methods that do not require the use of a gradient in picking the search direction are called for. Note that the same difficulty eliminates the use of steepest descent methods.

A DIRECTION SET METHOD

A method which does not require knowledge of the gradient, and which takes advantage of the quadratic nature of the cost is as follows:

1. Pick mp orthogonal directions in the v_j space. For each of these directions in succession, perform the line search as in the next steps, taking advantage of the quadratic nature of the cost to find the true minimum in three repetitions.

2. Take a step along the chosen direction in v_j space, and apply the resulting control in the next repetition. Evaluate f from the data from this repetition.

3. If f decreased pick another step in the same direction, if it increased pick a step in the opposite direction, and apply to the system.

4. Since the quadratic cost surface in the plane obtained by the chosen direction is a parabola, the data from 2 and 3 determines this parabola, and can be used to find the minimizing v_j for this direction. This completes the line search.

5. Return to 2 with the next direction.

This algorithm will converge in 3mp+1 repetitions, provided there is no noise in the measurements. Further repetitions would be needed to eliminate noise effects. The algorithm has the desirable property that at the end of each line search involving three samples, the system performance is improved. Care must be taken to avoid large disturbances to the repetitive process during the line search.

AN IDENTIFICATION TEST METHOD

It is always possible to identify a system from impulse response tests. In this section we study the use of such tests, and then in the next section we will study how to accomplish improvement with each repetition. If data is taken for a sequence of repetitions (j=0,1,2,...,N) then equation (3) gives

$$\left[\delta_N y\ \ \delta_{N-1} y\ \ \cdots\ \ \delta_1 y\right] = P\left[\delta_N u\ \ \delta_{N-1} u\ \ \cdots\ \ \delta_1 u\right] \qquad (8)$$

Once N is sufficiently large, this can be solved for P by use of an inverse or pseudo-inverse. If one makes each $\delta_j \mathbf{u}$ a different column of the identity matrix, which is the discrete equivalent of an impulse response test, then one can eliminate the need for inverting a matrix. In other cases, one can take advantage of the lower block triangular nature of P to efficiently compute the inverse. Noise level can be important in deciding the best approach. Once P is determined, then solving (6) gives

$$\delta_{j+1}\mathbf{u} = \mathbf{v}_j = (P^T P)^{-1} P^T (\mathbf{y}_j - \mathbf{y}_D) \tag{9}$$

In the case that the desired trajectory is feasible for system 1, and the desired trajectory exactly specifies all the freedom in the system, then the solution to (9) is the same as solving (3) for $\mathbf{v}_j$ to obtain $\mathbf{v}_j = P^{-1} (\mathbf{y}_j - \mathbf{y}_D)$. This approach requires mp+1 steps to optimize the tracking in the no noise case.

A MODIFICATION TO OBTAIN IMPROVEMENT EACH REPETITION

In practice one usually has a model of the system, call it P_m, when one starts the repetitive control, although one may not be very confident about the model. As the data from each repetition arrives one could modify the model to match the data, using the smallest possible change in the elements in the least squares sense. At step j, one would choose ΔP to satisfy

$$\left[\delta_j \mathbf{y} \;\; \delta_{j-1}\mathbf{y} \;\; \cdots \;\; \delta_1 \mathbf{y}\right] = \left[P_m + \Delta P\right]\left[\delta_j \mathbf{u} \;\; \delta_{j-1}\mathbf{u} \;\; \cdots \;\; \delta_1 \mathbf{u}\right]$$

or

$$\Delta P = \left\{\left[\delta_j \mathbf{y} \;\; \cdots \;\; \delta_1 \mathbf{y}\right] - P_m \left[\delta_j \mathbf{u} \;\; \cdots \;\; \delta_1 \mathbf{u}\right]\right\}\left[\delta_j \mathbf{u} \;\; \cdots \;\; \delta_1 \mathbf{u}\right]^+ \tag{10}$$

where the superscript + indicates the Moore-Penrose pseudoinverse. One can show that this pseudoinverse will minimize $tr(\Delta P^T \Delta P)$, i.e. the sum of the squares of the changes of all elements of ΔP.

The method becomes:

1. Pick a direction for $\delta_1 \mathbf{u}$ and make the first repetition to obtain the $\delta_1 \mathbf{y}$, $\delta_1 \mathbf{u}$ pair.

2. Solve for ΔP from (10).

3. Obtain $\delta_{j+1}\mathbf{u}$ from (9) substituting $P_m + \Delta P$ for P. The singular value decomposition of $[\delta_{j+1}\mathbf{u} \;\; \cdots \;\; \delta_1 \mathbf{u}]$ involved in the pseudo-inverse will determine if $\delta_{j+1}\mathbf{u}$ is linearly independent. In the event that it is not, choose a linearly independent direction at this step.

4. Repeat steps 2 and 3 until convergence is obtained.

This approach takes full advantage of a priori information about the system, improves the control system behavior with every repetition, handles noise in a natural way that smooths its influence, handles in a logical manner specified desired trajectories that are not feasible for the system to perform, and in the no noise case produces convergence in mp+1 repetitions or less. The effectiveness of the method by comparison to the direction set method above is fundamentally related to the fact that the quadratic cost (5) is not a general quadratic function, but rather the linear and the quadratic matrix coefficients P^T and P^TP are related, and this extra information has been incorporated in the new algorithm. Rather than trying to identify the quadratic surface of f as in the direction set method, the linear relation (3) from which the quadratic is derived is identified, with a resulting savings in repetitions. A numerical optimization method with this same property is presented in the next section.

The above method involves identification of the system model in the repetition domain, which by analogy to learning control can be called learning identification. Various approaches to identification in the repetition domain are presented in [7,9,10], for both time-varying and time-invariant cases.

When the system is time-invariant one can accelerate the learning control convergence by using the identification of elements of P to compute the time domain description A, B, C, from which one can complete the elements of P before enough data has been taken to obtain all elements directly. The algorithm is as follows:

1. While applying the above algorithm, use the lower block triangular nature of P to determine successively CB, CAB, CA^2B, etc.

2. Form the Hankel matrix of these Markov parameters for the appropriate dimension, if one knows the order n of the system, for any repetition number large enough to have generated $CA^{2n-1}B$. Later repetitions allow more smoothing. If n is not known, monitor the rank of the Hankel matrix with repetitions.

3. Use the Eigensystem Realization Algorithm (see for example [11], or use the Ho Kalman algorithm) to obtain A, B, and C. Note that one does not need to identify $w_j(k)$ of the time domain model.

4. Use A, B, and C to construct the full P matrix.

5. Find the optimizing control using this in (9).

Note that if the full state is measured, then the Eigensystem Realization Algorithm can be bypassed, by simply identifying A and B using a pseudoinverse solution of the sequence of equations (1) for succeeding steps, differenced from one repetition to the next to eliminate $w_j(k)$.

GENERALIZED SECANT METHOD WITH GAUSS-NEWTON OPTIMIZATION

The algorithm generated in the preceding section bears considerable similarity in philosophy to the generalized secant method of Barnes, and various modifications due to Fletcher and Broyden (see [8, pp.129-133], and [12-15]). These methods have computational advantages due to their attention to generating a recursive calculation and to use of the properties of rank 1 updates. The generalized secant method approximates P by a P_{j+1} computed from

$$P_{j+1} = P_j + \frac{[y_{j+1} - y_j - P_j (v_{j+1} - v_j)]\, z_j^T}{z_j^T (v_{j+1} - v_j)} \qquad (11)$$

where the z_j are any vector orthogonal to $v_2 - v_1, v_3 - v_2, \ldots, v_j - v_{j-1}$. The resulting P_{mp+1} satisfies

$$P_{mp+1} (v_{j+1} - v_j) = (y_{j+1} - y_D) - (y_{j+1} - y_D) \qquad j=1,2, \ldots \qquad (12)$$

The approach allows the iteration to proceed beyond the mp+1 step in an efficient way, to eliminate effects of noise, by choosing z_{mp+k} to be orthogonal to the most recent mp differences in successive values of v. The identified P is used to find the minimizing v using the derivative condition (6) (i.e., a Gauss-Newton method of optimization could be used).

In order to make this method produce an improvement at each repetition, a pseudo-inverse can again be used. Certain algorithms are suggested to update the pseudoinverse, including updates in QR or LU decomposition to maintain accuracy.

CONCLUSION

The learning control methods developed here are more complicated than those of [4], but can produce guaranteed convergence. On the other hand, they can be simpler than the adaptive approaches of [5-7] which can guarantee convergence. The final approaches generated here develop intelligent ways to adjust the control action so as to steadily improve controller performance without the need for any test signals designed for identification purposes only. As such, the paper complements [9] in picking control actions which are "sufficiently general." The methods here may also have advantages in alleviating requirements on specifying desired trajectories that are feasible, as studied in [4].

This paper has given an overview of the possible numerical unconstrained optimization methods which might be used to generate learning control laws. The lack of knowledge of the derivatives of the performance criterion constrains the range of the algorithms. It was shown that faster convergence can be obtained by focusing on identification in the repetition domain followed by minimization, than can be obtained by a direct attempt to minimize the quadratic function of the error. Derivative free approaches such as the Secant/Gauss-Newton method are seen to represent recursive parameter identification routines for the repetition domain (similar to those developed in [9]). There are various numerical considerations to be investigated, for both speed of computation and accuracy, and including use of the lower block triangular nature of P, and these will be studied in a future work. Also, the application of the control methods discussed here will be studied as applied to manufacturing problems.

REFERENCES

1. S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering Operations of Robots by Learning," Journal of Robotic Systems, Vol. 1, No. 2, 1984, pp. 123-140.

2. R. H. Middleton, G. C. Goodwin, and R. W. Longman, "A Method for improving the Dynamic Accuracy of a Robot Performing a Repetitive Task," Technical Report No. EE8546, Department of Electrical and Computer Engineering, University of Newcastle, Newcastle, Australia 2308, 1985. To appear in the International Journal of Robotics Research.

3. E. G. Harokopos and R. W. Longman, "A Learning Controller for Tracking Arbitrary Inputs in Robot Motions," Proceeding of the Tenth IASTED International Symposium on Robotics and Automation, Lugano, Switzerland, June 29 - July 2, 1987.

4. M. Phan and R. W. Longman, "A Mathematical Theory of Learning Control for Linear Discrete Multivariable Systems," Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, Minneapolis, Minnesota, August 1988.

5. M. Phan and R. W. Longman, "Liapunov Based Model Reference Learning Control," Proceedings of the Twenty-Sixth Annual Allerton Conference on Communication, Control, and Computing, Allerton House, Monticello, Illinois, 1988.

6. M. Phan and R. W. Longman, "Indirect Learning Control with Guaranteed Stability," Proceedings of the 1989 Conference on Information Sciences and Systems, Johns Hopkins University, Baltimore, MD, 1989.

7. C. J. Li and H. S. M. Beigi, "A Self-tuning Regulator with Learning Parameter Estimation for Robot Manipulators," submitted for publication.

8. R. Fletcher, Practical Methods of Optimization, Second Edition, John Wiley & Sons, New York, 1987.

9. M. Phan and R. W. Longman, "Learning System Identification," Proceedings of the Modeling and Simulation Conference, Pittsburgh, PA, 1989, to appear.

10. J. N. Juang, L. Horta, and R. W. Longman, "ERA System Identification in the Repetition Domain," Proceedings of the 1989 AIAA/AAS Astrodynamics Specialist Conference, 1989, to appear.

11. R. W. Longman and J. N. Juang, "A Variance Based Confidence Criterion for ERA Identified Modal Parameters," Astrodynamics, 1987, Advances in the Astronautical Sciences, Vol. 65, Part 1, 1988, pp. 581-601.

12. J. G. P. Barnes, "An Algorithm for Solving Nonlinear Equations Based on the Secant Method," Computer Journal, Vol. 8, 1965, pp. 66-72.

13. C. G. Broyden, "A Class of Methods for Solving Nonlinear Simultaneous Equations," Mathematics and computer, Vol. 19, 1965, pp. 577-593.

14. R. Fletcher, "A Technique for Orthogonalization," Journal of the Institute for Mathematics Applications, Vol. 5, 1969, pp. 116-162.

15. R. Fletcher and S. P. J. Matthews, "A Stable Algorithm for Updating Triangular Factors Under a Rank One Change," Mathematics Comp., Vol. 45, 1985, pp. 471,485.