New Adaptive and Learning-Adaptive Control Techniques based on an Extension of the Generalized Secant Method

Homayoon S.M. Beigi
T.J. Watson Research Center
International Business Machines
P.O. Box 218
Yorktown Heights, New York 10598

Abstract

This paper presents two new controllers and uses an extension of the generalized secant method of solving linear equations. [1] The first controller is adaptive and operates in the time domain. The second one is a learning-adaptive controller which uses a similar mathematical concept, but it is formulated to deal with repetitive systems with repetitive disturbances. This controller operates in the repetition domain. For both controllers, the formulation of the systems into a linear estimation problem is presented as well as the solution using the secant method. A convergence is also given for both controllers. In addition, some practical modifications are presented to increase the robustness and stability of the systems. Furthermore, an analog network is proposed for increasing the speed of the controllers which is especially useful for the learning-adaptive controller since this network parallelizes most of the computation necessary for establishing the control, in hardware form. The adaptive controller is tested in simulations of two highly non-linear dynamic systems (a non-linear mass-spring-dashpot and a damped pendulum) against the performance of a self-tuning regulator and shows considerable superiority both in terms of speed of convergence and accuracy over the self-tuning regulator. The learning-adaptive controller also shows a great performance for controlling repetitive dynamic systems when applied to the same two non-linear systems, operating repetitively. Non-linearities of these dynamic systems have been amplified by requiring them to follow a highly non-linear and demanding trajectory.

1 Introduction

In general, most dynamic systems are too complex to be modeled properly. In most cases, an ideal control system would therefore be a system which gives the best performance for the least user-provided amount of information about the system being controlled. Adaptive control systems have generally been developed for such applications. Two well known adaptive controllers are the Model-Reference adaptive controllers and the Self tuning Regulator. [2] As its first proposal, this paper presents a solution to the adaptive-control problem by considering a general discrete-time, linear, time-invariant model given by the following equation,

$$y(t+1) = \underbrace{CA}_{\hat{A}} x(t) + \underbrace{CB}_{\hat{B}} u(t) + w(t) \tag{1}$$

In equation 1 the state x is n-dimensional, the control u is m-dimensional, the output y is q-dimensional, and t is the time step. Also, $\hat{A}$, $\hat{B}$, and $w(t)$ are assumed to be unknown – otherwise one could determine in advance what control to use to minimize the tracking error. For simplicity, the dimension (order) of the system is assumed to be known, but generalization to just knowing

an upper bound on the order is easily considered.

Many practical applications of control theory involve systems which are repeatedly asked to perform the same task. Examples include a large number of manufacturing problems as well as tracking problems for robots on an assembly line. Standard controller design methods often produce systems that do an imperfect job of executing a command and when these commands are repeated, the systems repeat the same errors every time. It is perhaps a bit primitive to persist in repeating the same errors. In the last few years a theory of learning control has been developed in the literature, generating controllers that can learn from their previous experience at performing a specific task, see [3, 4, 5, 6].

When control systems are given a task to perform repeatedly, they will usually repeat the same errors in executing the command, except for some random noise effects. Learning controllers on the other hand can improve their performance at a given task with each repetition of the task. Learning control algorithms have used many different approaches to the solving this problem, such as analogues of proportional, derivative and integral control in the repetition domain to eliminate errors [6, 5] or conversions of the adaptive control approaches to the learning control problem [3].

As its second proposal, this paper presents a new learning-adaptive control method which improves the performance of a dynamic system as the desired task is performed by the system over and over again. In this approach a time-variant, discrete-time model of the system is devised in the form of a system of linear algebraic equations relating the change in the state of the system to the change in the control action from one repetition of the task to the next. This set of linear equations gives the transition between any two repetitions. This system is then solved using a generalized secant method for the appropriate control action and parameter estimates that will minimize the tracking error of the controlled dynamic system, only requiring the availability of the order of the system, without any prior knowledge of the system parameters. For practical implementation, a parallel analog network is also presented which will speed up the most computation intensive part of the algorithm, even though this computation is done offline for each repetition, and it could even be done in between repetitions.

The next section presents the adaptive controller including its formulation and convergence. Then, the formulation of the learning-adaptive controller is presented along with its convergence proof. Later, some practical modifications are presented for the two systems including the parallelization of the computation in the learning-adaptive controller. Simulation results of the two controllers are then given in the following section, followed by a conclusion.

2 The Adaptive Controller

2.1 Problem Formulation

Let us assume that the elements in u(t) in equation 1 could be arranged in such a way that all the elements which are dependent on time t are placed at the bottom portion of the vector u(t) and all the elements which signify the effects of the previous $0 \leq \tau < t$ inputs are placed on the

top portion of the $u(t)$ vector. Namely,

$$u(t) = \left[\underbrace{u^{1^T}(t-1)}_{1xm_1} | \underbrace{u^{2^T}(t)}_{1xm_2} \right]^T \tag{2}$$

Therefore, equation 1 may be rewritten in the following form,

$$y(t+1) = \hat{A}x(t) + \underbrace{\left[\hat{B}^1|\hat{B}^2\right]}_{\hat{B}} \begin{bmatrix} u^1(t-1) \\ u^2(t) \end{bmatrix} + w(t) \tag{3}$$

Rewrite equation 3 as follows,

$$y(t+1) = \underbrace{\begin{bmatrix} \hat{A} & \hat{B} \end{bmatrix}}_{S} \underbrace{\begin{bmatrix} x(t) \\ u(t) \end{bmatrix}}_{v(t)} + w(t) \tag{4}$$

Let us assume for now that there is some control action $u^2(t)$ such that when applied to the real system, S, it would generate an output y_{t+1} equal to the desired value y^d_{t+1} and let us call the v vector containing that control, $\tilde{v}_t$. Therefore,

$$y^d_{t+1} = S\tilde{v}_t \tag{5}$$

Now let us assume that there exists some combination of the system matrix S and the control input $u^2(t)$ such that when placed in the v vector, it provides the same result as in equation 5. Let us call this pair S_t and v_t. Therefore,

$$y^d_{t+1} = S_t v_t \tag{6}$$

There might not exist such a pair of S and v that would get the output at time $t+1$ to be equal to the desired value. However, given an S_t, one could always find some $u^2(t)$ such that when placed in v, the Frobenius norm of the difference between the actual output and the desired output at time $t+1$ is minimized. This $u^2(t)$ is given by the following equation:

$$u^2(t) = \hat{B}_t^{2^\dagger} \left(y^d(t+1) - \left[\hat{A}_t|\hat{B}_t{}^1\right] \begin{bmatrix} x(t) \\ u^1(t-1) \end{bmatrix} \right) \tag{7}$$

$u^2(t)$ given by equation 7 minimizes the Frobenius norm of the error, $||y^d(t+1) - y(t+1)||_{\mathcal{F}}$.

Through the application of the control given by equation 7, to the actual system, at time t, some output y_{t+1}, will be generated. Let us assume that at time t, the best estimate of the system matrix available to us is S_t. Therefore, the difference between the true system matrix S and S_t could be denoted by ΔS_t. Now let us impose the condition that S_{t+1} would generate the same output y_{t+1} as would the real system, S, with the given v_t. Therefore,

$$y_{t+1} = S_{t+1} v_t \tag{8}$$

Rewriting equation 8 in terms of S_t and the difference between S_t and S_{t+1} such that equation 8 holds,

$$y_{t+1} = (S_t + \Delta S_t)\, v_t \tag{9}$$

One way to solve equation 9 is to update the system matrix in one direction each time using the generalized secant update [1] in the following fashion,

$$S_{t+1} = S_t + \Delta S_t \tag{10}$$

where,

$$\Delta S_t = \frac{(y(t+1) - S_t v_t)\, z_t^T}{z_t^T v_t} \tag{11}$$

z_t's are secant projection vectors, chosen in the following way:
- *If* $t \geq n+m-1$, *then,* z_t *is chosen orthogonal to the previous* $n+m-1$ *vectors,* $v_{t-(n+m-1)}, \cdots, v_{t-1}$.
- *If* $t < n+m-1$, *then,* z_t *is chosen orthogonal to the available* t *vectors,* $v_0, \cdots, v_{t-1}$.

One possibility is to pick z_t as a linear combination of $v_0, \cdots, v_t$ which would be orthogonal to all $v_0, \cdots, v_{t-1}$. A few orthogonalization methods are available in the literature which may be used for the actual evaluation of the z vectors. These methods include the well-known Gram-Schmidt orthogonalization process [7] and a more advanced technique due to Fletcher [8] in which the number of vectors in the set are likely to be increased or decreased. Therefore, using an orthogonalization method and equations 7 and 10, in a recursive fashion, a control could be evaluated using equation 7 and after being applied to the real system, based on the response of the system, a better estimate of the system matrices may be obtained using equation 10.

2.2 Convergence

The set of z's picked in equation 10 has the property that,

$$\Delta S_t v_j = 0 \qquad 0 \leq t - j < n + m - 1 \tag{12}$$

therefore,

$$S_{t+1} v_j = [S_{t+1} + \Delta S_{j+1} + \cdots + \Delta S_t]\, v_j \tag{13}$$

$$S_{t+1} v_j = S_{j+1} v_j \qquad 0 \leq t - j < n + m - 1 \tag{14}$$

Now assume that some $s \leq n + m$ is the maximum number of linearly independent v's which could be found and thus v_s is a linear combination of all previous s linearly independent vectors.
Then,

$$v_s = \alpha_0 v_0 + \cdots + \alpha_{s-1} v_{s-1} \tag{15}$$

From equation 15,

$$S_s v_s = \sum_{j=0}^{s-1} \alpha_j S_s v_j \tag{16}$$

Also,

$$S_{i+1}v_j = S_{j+1}v_j \;\; 0 \leq j \leq s-1 \; (from \; eq. \; 14) \tag{17}$$

$$S_{i+1}v_j = y_{j+1} \quad (from \; eq's \; 9 \; and \; 10) \tag{18}$$

$$S_{i+1}v_j = Sv_j \quad (from \; eq. \; 4) \tag{19}$$

Substituting equation 19 into 16,

$$S_s v_s = S \sum_{j=0}^{s-1} \alpha_j v_j \tag{20}$$

or,

$$S_s v_s = S v_s \tag{21}$$

Using equations 4 and 21 in the absence of noise,

$$\begin{aligned} y_{s+1} &= S_s v_s \\ &= S v_s \end{aligned} \tag{22}$$

Therefore, after $s \leq n+m$ time instants, the control given by equation 7 provides an output which will minimize the Frobenius norm of the error at time $s+1$ (i.e., $min||y^d(s+1) - y(s+1)||_{\mathcal{F}}$).

3 The Learning-Adaptive Controller

3.1 Problem Formulation

Consider a general discrete-time linear time-variant or time-invariant system,

$$x^k(t+1) = A(t)x^k(t) + B(t)u^k(t) + w^k(t) \tag{23}$$

$$\begin{aligned} y^k(t+1) &= C^k(t+1)x^k(t+1) \\ t &= 0, 1, 2, \ldots, p-1 \\ k &= 0, 1, 2, \ldots \end{aligned} \tag{24}$$

where the state x is n-dimensional, the control u is m-dimensional, the output y is q-dimensional ($q \geq m$), t is the time step in a p-step repetitive operation, and k is the repetition number. Also, A, B, C, and $w^k(t)$ are assumed to be unknown – otherwise one could determine in advance what control to use to minimize the tracking error. For simplicity, the dimension n of the system is assumed known, but generalization to just knowing an upper bound on n is easily considered.

In the learning control problem (as contrasted with the repetitive control problem in [5]) 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^k is reserved for the signal added to the control for learning purposes. A time-variant model is considered

because many applications such as in robotics and machining involve nonlinear dynamic systems which, when linearized, produce linear models with coefficients that vary with the time step and vary in the same manner each repetition. In such repetitive operations it is often the case that there will be disturbances $w^k(t)$ 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 a feedback control is to handle any non-repetitive disturbances, and these will be ignored for purposes of designing the learning controller.

The solution to 23 can be written as,

$$x^k(t+1) = \left(\prod_{j=0}^{t} A(j)\right) x^k(0) + \sum_{j=0}^{t} \left(\left(\prod_{r=j+1}^{t} A(r)\right) (B(j)u^k(j) + w^k(j))\right) \tag{25}$$

where the product symbol is taken to give the identity matrix if the lower limit is larger than the upper limit. Let p be the total number of time steps in one repetition of a given task.

Writing equation 25 for successive repetitions k and $k+1$ and subtract the equation for repetition k from that of repetition $k+1$. Assuming that disturbances w^k are repetitive with a period of p or a period of any integral divisor of p, and that the initial conditions are also repetitive from repetition k to $k+1$, we may write the following relation for the outputs of two consecutive repetitions,

$$\mathbf{y}^{k+1} - \mathbf{y}^k = P \underbrace{(\mathbf{u}^{k+1} - \mathbf{u}^k)}_{\mathbf{v}^k} \tag{26}$$

$$P = \begin{bmatrix} C(1)B(0) & 0 & \cdots & 0 \\ C(2)A(1)B(0) & C(2)B(1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C(p)A(p-1)\cdots A(1)B(0) & \cdots & \cdots & C(p)B(p-1) \end{bmatrix} \tag{27}$$

The system transition matrix P is given by equation 27, and,

$$\mathbf{y}^k = \begin{bmatrix} y^{kT}(1) & y^{kT}(2) & \cdots & y^{kT}(p) \end{bmatrix}^T \tag{28}$$

$$\mathbf{u}^k = \begin{bmatrix} u^{kT}(0) & u^{kT}(1) & \cdots & u^{kT}(p-1) \end{bmatrix}^T \tag{29}$$

Defining the error $\mathbf{e}^k \equiv \mathbf{y}^k - \mathbf{y}_d$ where $\mathbf{y}_d$ is the discrete desired trajectory, equation 26 can be written as,

$$P\mathbf{v}^k = \mathbf{e}^{k+1} - \mathbf{e}^k \tag{30}$$

At any repetition k, suppose there exists some change in our input vector, $\tilde{\mathbf{v}}^k$, which would lead to a zero tracking error at the next repetition, $k+1$, namely, $\mathbf{e}^{k+1} = \mathbf{0}$. Then, equation 30 for the given change of input would be as follows,

$$P\tilde{\mathbf{v}}^k = \mathbf{0} - \mathbf{e}^k \tag{31}$$

If P^k is an approximation to P at the k^{th} repetition, then one can write,

$$P^k\mathbf{v}^k = -\mathbf{e}^k \tag{32}$$

where $\mathbf{v}^k$ is a change in the control vector at repetition k that would use the information in the approximation to P (P^k) and it would result in an $\mathbf{e}^{k+1}$ which is zero (or is minimum $\mathcal{F}$ norm as will be discussed later.)

Let us assume that we are provided with an initial guess for the P matrix at the zeroth repetition,

$$P^0\mathbf{v}^0 = -\mathbf{e}^0 \tag{33}$$

then, $\mathbf{v}^0$ may be solved for, by using the error vector $\mathbf{e}^0$ provided by the real system using control vector $\mathbf{u}^0$.

Generally, an exact $\mathbf{v}^0$ may not exist, but a minimum error solution may be obtained for $\mathbf{v}^0$ through using the Moore-Penrose pseudo-inverse in the sense that $\mathbf{v}^0$ would minimize,

$$\| P^0\mathbf{v}^0 + \mathbf{e}^0 \|_{\mathcal{F}} \tag{34}$$

namely,

$$\mathbf{v}^0 = -P^{0\dagger}\mathbf{e}^0 \tag{35}$$

If we keep 32 satisfied for all k, then we may write,

$$\mathbf{v}^k = -P^{k\dagger}\mathbf{e}^k \qquad for\ all\ k \tag{36}$$

minimizing $\| P^k\mathbf{v}^k + \mathbf{e}^k \|_{\mathcal{F}}$.

At any repetition k, the actual system parameters, P, could be written as,

$$P = P^k + D^k \tag{37}$$

where D^k is a matrix of corrections for P^k at each repetition k. Substituting for P in equation 30 from 37,

$$\left(P^k + D^k\right)\mathbf{v}^k = \mathbf{e}^{k+1} - \mathbf{e}^k \tag{38}$$

Solve for $D^k\mathbf{v}^k$ from equation 38,

$$D^k\mathbf{v}^k = \mathbf{e}^{k+1} - \mathbf{e}^k - P^k\mathbf{v}^k \tag{39}$$

Since $\mathbf{e}^{k+1}$ is the error through the introduction of $\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{v}^k$, then the only unknown in equation 39 is the correction matrix D^k.

Since D^k is a generally a matrix, one solution to equation 39 would be,

$$D^k = \frac{\left(\mathbf{e}^{k+1} - \mathbf{e}^k - P^k\mathbf{v}^k\right)\mathbf{z}^{k^T}}{\mathbf{z}^{k^T}\mathbf{v}^k} \tag{40}$$

where $\mathbf{z}^k$ are the secant projection vectors and are chosen such that, [1]
- *If* $k \geq mp-1$, *then,* $\mathbf{z}^k$ *is chosen orthogonal to the previous* $mp-1$ *control steps,* $\mathbf{v}^{k-(mp-1)}, \cdots, \mathbf{v}^{k-1}$.
- *If* $k < mp-1$, *then,* $\mathbf{z}^k$ *is chosen orthogonal to the available* k *steps,* $\mathbf{v}^0, \cdots, \mathbf{v}^{k-1}$.

One possibility is to pick $\mathbf{z}^k$ as a linear combination of $\mathbf{v}^0, \cdots, \mathbf{v}^k$ which would be orthogonal to all $\mathbf{v}^0, \cdots, \mathbf{v}^{k-1}$. Similar to the case of the adaptive control discussed earlier, any orthogonalization technique may be used for the actual evaluation of the $\mathbf{z}$ vectors.

With the above formulation in mind, a recursive generalized secant learning adaptive control scheme is given by the following two recursive equations and the use of an orthogonalization method such as Gram-Schmidt:

$$\mathbf{v}^k = -P^{k\dagger}\mathbf{e}^k \tag{41}$$

$$P^{k+1} = P^k + \frac{\left(\mathbf{e}^{k+1} - \mathbf{e}^k - P^k\mathbf{v}^k\right)\mathbf{z}^{k^T}}{\mathbf{z}^{k^T}\mathbf{v}^k} \tag{42}$$

3.2 Convergence

The set of $\mathbf{z}^k$ picked in equation 42 has the property that

$$D^k\mathbf{v}^j = \mathbf{0} \qquad 0 \leq k - j < mp - 1 \tag{43}$$

therefore,

$$\begin{aligned} P^{k+1}\mathbf{v}^j &= \left[P^{j+1} + D^{j+1} + \cdot + D^k\right]\mathbf{v}^j \\ &= P^{j+1}\mathbf{v}^j \qquad 0 \leq k - j < mp - 1 \end{aligned} \tag{44}$$

Now assume that some $n \leq mp$ is the maximum number of linearly independent $\mathbf{v}$'s which could be found and thus $\mathbf{v}^n$ is a linear combination of all previous n linearly independent control steps. Then,

$$\mathbf{v}^n = \alpha_0\mathbf{v}^0 + \cdots + \alpha_{n-1}\mathbf{v}^{n-1} \tag{45}$$

From equation 45,

$$P^n\mathbf{v}^n = \sum_{j=0}^{n-1} \alpha_j P^n\mathbf{v}^j \tag{46}$$

Also,

$$P^{i+1}\mathbf{v}^j = P^{j+1}\mathbf{v}^j \; 0 \leq j \leq n-1 \; (from\ eq.\ 44) \tag{47}$$

$$P^{i+1}\mathbf{v}^j = \mathbf{e}^{j+1} - \mathbf{e}^j \qquad (from\ eq's\ 38\ and\ 42) \tag{48}$$

$$P^{i+1}\mathbf{v}^j = P\mathbf{v}^j \qquad (from\ eq.\ 30) \tag{49}$$

Substituting equation 49 into 46,

$$P^n\mathbf{v}^n = P\sum_{j=0}^{n-1} \alpha_j\mathbf{v}^j \tag{50}$$

or,

$$P^n\mathbf{v}^n = P\mathbf{v}^n \tag{51}$$

Using equations 30, 38 and 51 in the absence of non-repetitive noise,

$$\begin{aligned}\mathbf{e}^{n+1} - \mathbf{e}^n &= P^n \mathbf{v}^n \\ &= P\mathbf{v}^n \end{aligned} \tag{52}$$

If $\mathbf{v}^n$ satisfies equation 32, then,

$$\mathbf{e}^{n+1} - \mathbf{e}^n = -\mathbf{e}^n \tag{53}$$

which suggests that $\mathbf{e}^{n+1}$ must be zero. However, if $\mathbf{v}^n$ only satisfies equation 32 in least $\mathcal{F}$-norm sense, then $\mathbf{e}^{n+1}$ would have a minimum $\mathcal{F}$-norm, namely,

$$\mathbf{e}^{n+1} = P\mathbf{v}^n + \mathbf{e}^n \tag{54}$$

where $\mathbf{e}^{n+1}$ has minimum $\mathcal{F}$-norm.

If n (the rank of $P^0 - P$ is equal to mp, then after $mp + 1$ repetitions of the task, a minimum $\mathcal{F}$-norm tracking error will be achieved. Of course, n, the rank of $P^0 - P$ may be less than mp in which case if the directions in P^0 containing the correct portions of P are given, then n would be less than mp by that number of correct guesses. However, if one is not sure of his initial guess for the P matrix, then $mp + 1$ is the maximum number of repetitions before convergence.

4 Practical Parallel Implementation

This section provides two different suggestions for using the above two algorithm in a practical setting. The first suggestion has shown to be necessary for preserving robustness and stability in both controllers. The second suggestion is a parallel implementation of the pseudo-inverse step which proves to be the most computation intensive part of the algorithms. It is most useful for the learning-adaptive controller.

4.1 Degree of Linear Independence

As a practical precaution for stability and quick convergence after evaluating the vector v_τ using equation 7 in the adaptive case (or equation 41 in the learning-adaptive case), a linear independence test was made by evaluating,

$$\omega_\tau = \frac{|z_\tau^T v_\tau|}{||z_\tau||_\mathcal{E}||v_\tau||_\mathcal{E}} \tag{55}$$

Here, τ should be replaced with t for the adaptive controller and with k for the learning-adaptive controller. If

$$\omega_\tau < \rho \quad 0 < \rho < 1 \tag{56}$$

(In the simulations of this paper, $\rho = 0.0001$), then the control for that step is changed such as to make $\omega_\tau > \rho$. This is done using the information provided by the orthogonalization method. [7]

4.2 Parallel Implementation

The most computation intensive part of the learning-adaptive algorithm is the evaluation of the pseudo-inverse in equation 41. Although, the nature of this computation is such that it could be done between two repetitions, it would be nice to be able to solve equation 41 in a more effective and quicker fashion. By nature, pseudo-inversion is a minimization problem such that the Frobenius norm of the error $||P^k\mathbf{v}^k + \mathbf{e}^k||_{\mathcal{F}}$ is minimized. This error, E, could also be formulated in terms of a minimization problem in $\mathbf{v}^k$,

$$\frac{d\mathbf{v}^k}{d\xi} = -M(\xi)\nabla E(\mathbf{v}^k) \tag{57}$$

$$\nabla E(\mathbf{v}^k(\xi)) = {P^k}^T(P^k\mathbf{v}^k(\xi) + \mathbf{e}^k) \tag{58}$$

$$\mathbf{v}^k(0) = \mathbf{v}_0^k \tag{59}$$

This is a possible minimization formulation of the pseudo-inverse. If the weighting factor $M(\xi)$ is set to the identity matrix, the minimization problem reduces to a steepest descent problem. For convergence of the minimization problem, M should be chosen to be positive definite. Reference [10] shows a network which recursively goes through the calculations of equations 57- 59. Figure 1 shows a parallel network which basically solves this minimization problem. However, since this is a parallel analog network, the calculations are done much more quickly.

A proof of convergence of equations 41 and 42 is given above. This proof of convergence is independent of the solution for the change in the control vector. The change in the control input vector converges if M is picked to be positive definite. This proof could be found in any mathematical handbook or reference [10] Therefore, the two problems are almost decoupled and if both converge, the system will converge.

5 Simulations and Results

Adaptive and Learning-Adaptive control algorithms based on the rectangularized version of the generalized secant method were tested on controlling two nonlinear dynamic systems via computer simulation. System 1 was a non-linear mass-spring-dashpot with the following differential equation:

$$m\ddot{\alpha} + c\,(1 + |\alpha|)\,\dot{\alpha} + (k|\alpha|)\,\alpha = T \tag{60}$$

System 2 was a damped pendulum with the following differential equation:

$$ml^2\ddot{\alpha} + c\dot{\alpha} + mglsin\alpha = T \tag{61}$$

where α in equation 60 is a measure of distance which is the linear position of the mass m from its equilibrium position. In equation 61, α is a measure of the angle the pendulum makes with the vertical axis (equilibrium point).

In the first system, $m = 0.1kgr$, $c = 0.1N/(m/s)$ and $k = 0.1N/m$. For the Pendulum, $m = 0.1kgr$, $c = 1N/(m/s)$, and $l = 0.1m$.

Both systems were given a very demanding non-linear trajectory to follow. This trajectory is given in figure 2 and it will ensure that both systems operate in a very highly non-linear region. This is a very good test of the robustness of the algorithms.

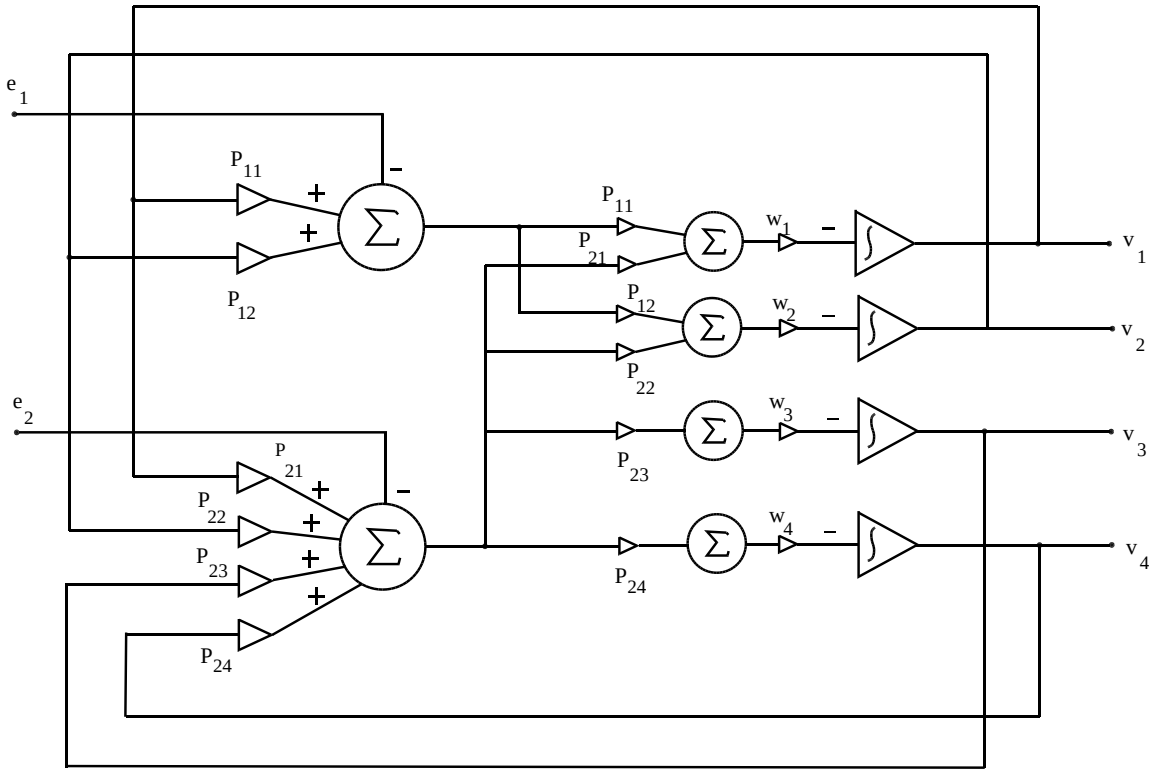

Figure 1: Parallel Network for Solving the Change in control $\mathbf{v}^k$

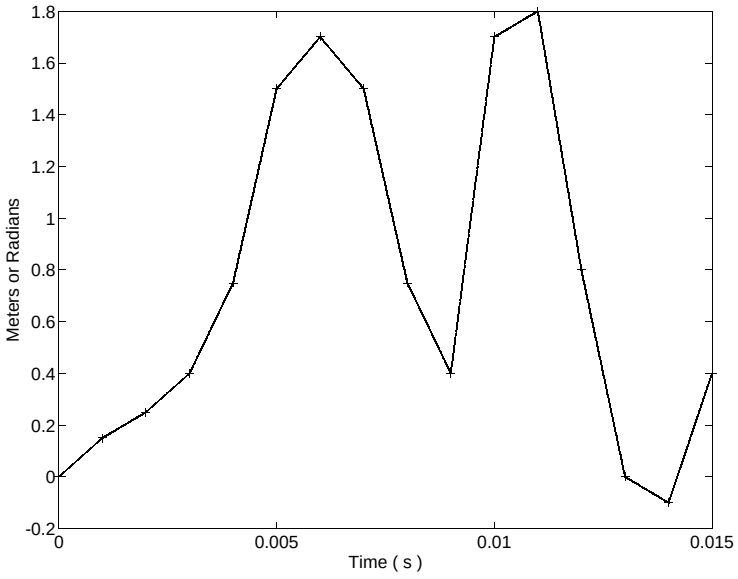

Figure 2: Desired Trajectory

5.1 The Adaptive Controller

The adaptive control algorithm of equations 7 and 10 was applied to controlling the noted non-linear systems. The performance of the adaptive-control algorithm is compared to that of the self-tuning regulator. In both the self-tuning regulator and the secant controller, a proportional-derivative (PD) controller was used to start up the control process. In the tested dynamic systems due to being second order, the first two time steps used the PD controller output which was then switched to the corresponding adaptive controller at time step 3. Both control processes started with the same initial guesses for the system parameters.

Figure 3 compares the secant controller with the self-tuning regulator when applied to equation 60. Figure 5 makes a similar comparison for the pendulum of equation 61. Figures 4 and 6 show comparison of the sum of squares of errors and Euclidean Norm of the Input Vector for the simulation runs of figures 3 and 5 respectively. Other runs were made using different parameters for the above systems and the outcomes of the performances were very consistent with those presented here. As it is apparent from figures 4 and 6, not only was the error reduced by the Secant Adaptive Controller, but it also used less energy in terms of the magnitude of the input to achieve such results. For the non-linear mass-spring-dashpot system the amount of enery used by the Secant Adaptive Controller is one order of magnitude lower than that used by the Self-Tuning Regulator.

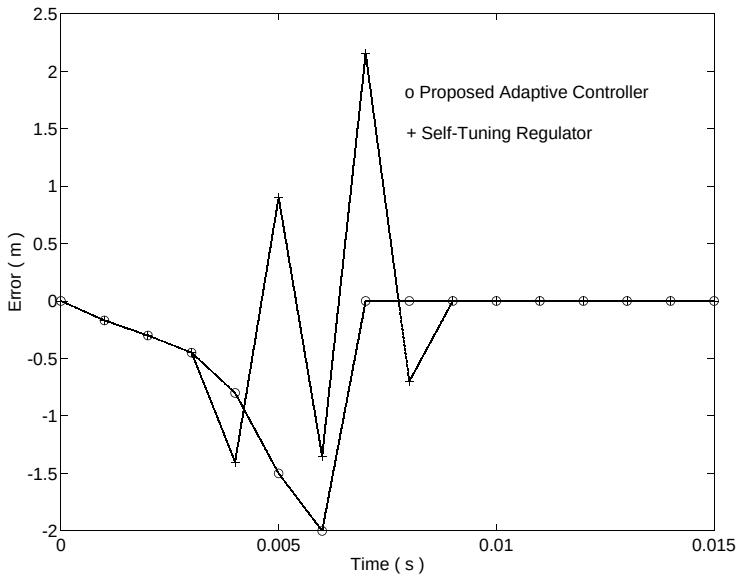

Figure 3: Non-linear Mass-Spring-Dashpot Output

5.2 The Learning-Adaptive Controller

Figures 7 shows the results of applying the learning-adaptive algorithm to system 1 without using rejection techniques of equation 55. It can be seen that convergence did not occur within the theoretically predicted 10 repetitions. However, by using the rejection technique, this problem is solved. Figures 8 and 9 show the sum of squares of errors in the first few repetitions of the two tasks. Through the use of rejections, the system observes the system passively and estimates the system parameters and as soon as some confidence level given by equation 55 is reached, it reduces the error practically to zero. The fact that both the simulated systems are highly nonlinear shows

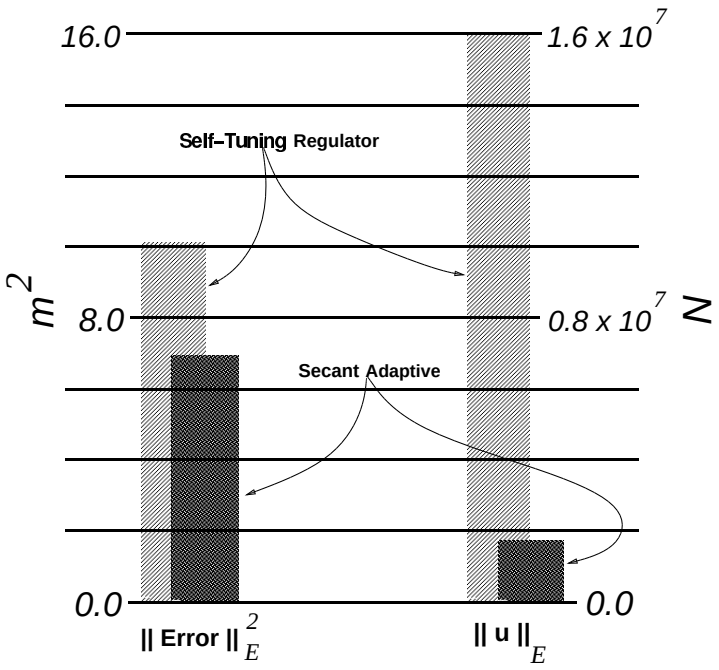

Figure 4: $||Error(t)||_E^2$ and $||u(t)||_E^2$ for the Non-Linear Mass-Spring-Dashpot System

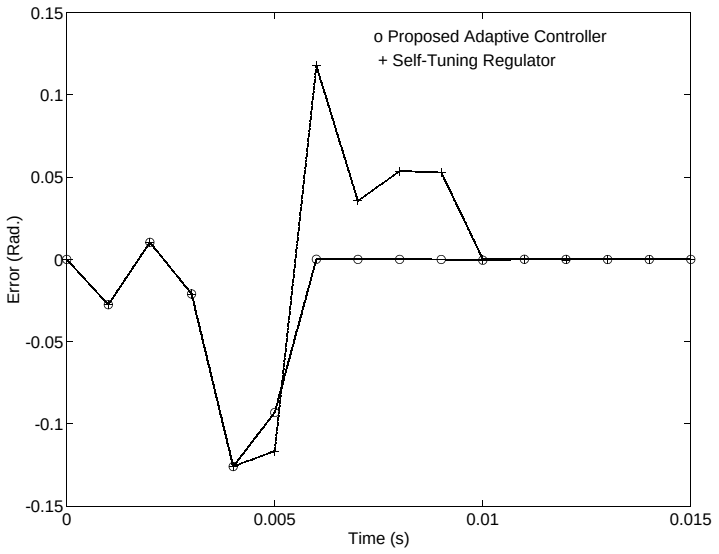

Figure 5: Pendulum Output

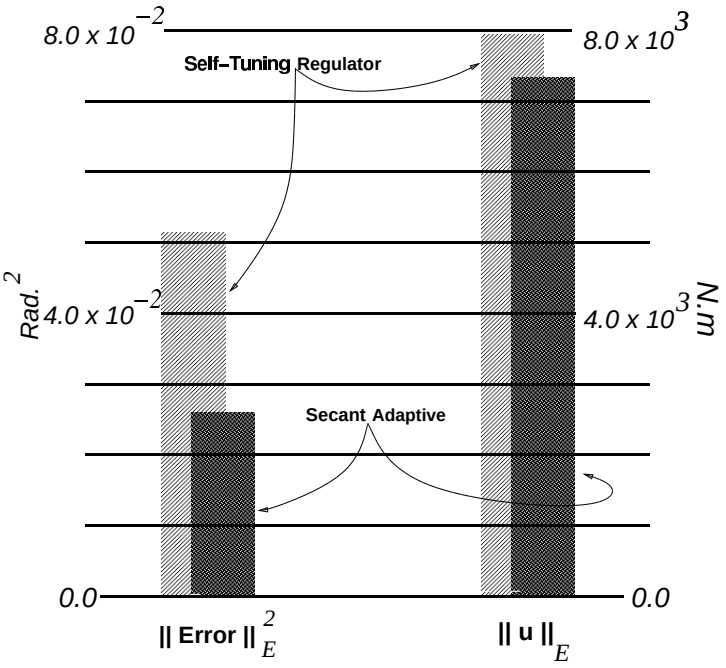

Figure 6: $||Error(t)||_E^2$ and $||u(t)||_E^2$ for the Pendulum

that this is a very robust learning-adaptive algorithm.

6 Conclusion

6.1 The Adaptive Controller

The secant adaptive-controller converges much more quickly than the self-tuning regulator and has a consistently lower overall error in the sum of squares sense (up to 65% lower). It is very robust to have been applied to such non-linear systems at highly non-linear trajectories displaying a very good performance and does not require any more computation than the self-tuning regulator. With the practical introduction of the restrictions on ω_t of equation 55, the secant controller takes a few steps to learn the parameters of the system before it greatly affects the dynamics of it. Once it realizes the system parameters, it quickly reduces the error to an amount very close to zero. This observation was done with many different systems simulated and also in the case of secant learning controller [7].

6.2 The Learning-Adaptive Controller

Theoretical evidence shows that the Generalized Secant method requires the least number of repetitions for convergence in this framework. [4] shows that through minimizing a quadratic cost function based on the sum of squares of the errors of the system, the fastest converging minimization technique will in general need an order of $(mp)^2$ repetitions for convergence. The method presented here will theoretically converge in at most (mp) repetitions in the absence of non-repetitive noise (as defined earlier).

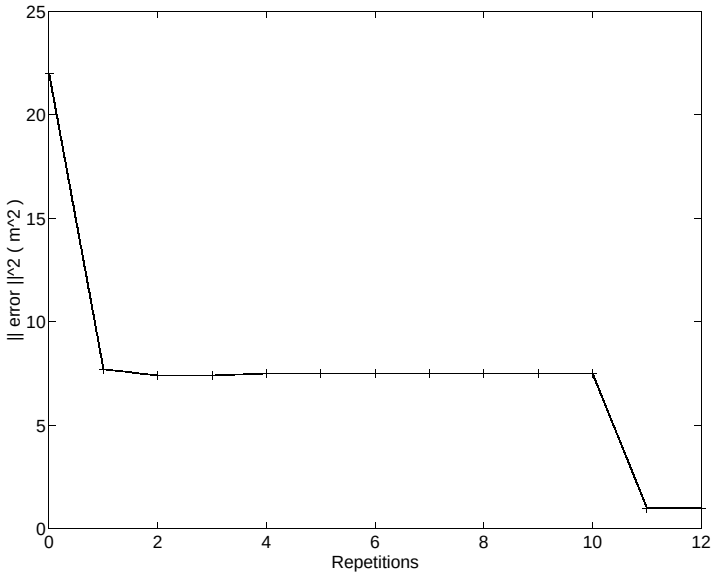

Figure 7: Learning-Adaptive on NLMSD WITHOUT Rejections

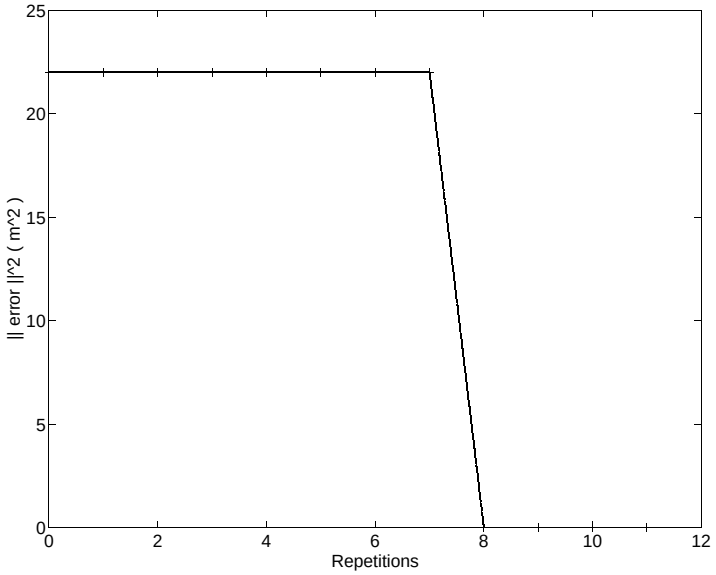

Figure 8: Learning-Adaptive on NLMSD WITH Rejections

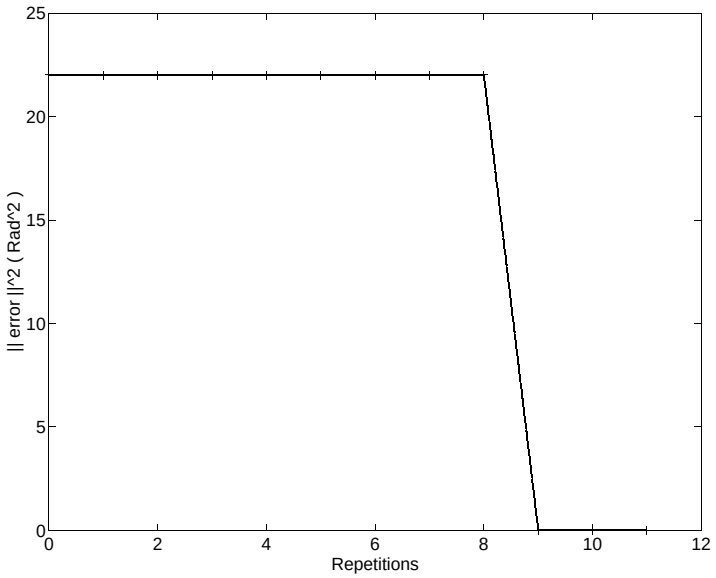

Figure 9: Learning-Adaptive on the Pendulum WITH Rejections

Results show that without rejection of non-informative control steps, the system may become unstable and that the use of a ρ which is too small in inequality 56, could result in a big reduction in the rate of convergence. Therefore, there is a trade-off in the magnitude of ρ. Also, in practice the magnitude of steps which replace rejected control steps should be made small enough to be considered a perturbation which does not cause instabilities in the system. In other words, these perturbations should be small enough that the standard stabilizing controller of the system would be able to handle. In general, the Generalized Secant Learning-Adaptive Controller has shown to be very stable and robust in a practical sense when applied to the nonlinear systems of this paper.

The generalized secant learning-adaptive controller was also combined with a parallel scheme for evaluating the pseudo-inversion required in that learning control algorithm. This makes the waiting period between two consequent repetitions shorter. The convergence of the two methods is shown to be almost independent. The results of simulations support the fact that using this parallel network for obtaining the pseudo-inverse is very practical. However, there is one problem of practicality which is still outstanding and that is the fact that for each time step there should be a connection built in the network. However, if networks are developed for this type of a controller, a maximum size network can work for any number of time steps less than or equal to its maximum capacity.

References

[1] J.G.P. Barnes, "An Algorithm for Solving Non-linear Equations Based on the Secant Method," Computer Journal, Vol. 8, pp. 66-72, 1965.

[2] K.J. Åström, "Adaptive Feedback Control," Proceedings of the IEEE, Vol. 75, No. 2, February 1987.

[3] C. James Li, Homayoon S.M. Beigi, Shengyi Li, Jiancheng Liang, "Nonlinear Piezo-Actuator Control by Learning Self-Tuning Regulator," ASME Transactions, Journal of Dynamic Systems, Measurement, and Control, Vol. 115, No.4, December 1993, pp. 720-723.

[4] R. W. Longman, H. S. M. Beigi and C. James Li, "Learning Control by Numerical Optimization Methods," Proceedings of Modeling and Simulation, Instrument Society of America, Vol. 20, Pat 5, 1989, pp.1877-1882.

[5] R.H. Middleton, G.C. Goodwin, and R.W. Longman, "A Method for Improving the Dynamic Accuracy of a Robot Performing a Repetitive Task," The International Journal of Robotics Research, Vol. 8, No. 5, October 1989, pp. 67-74.

[6] 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.

[7] Homayoon S.M. Beigi, "Neural Network Learning and Learning Control Through Optimization Techniques", Doctoral Thesis, Columbia University, 1991.

[8] R. Fletcher, "A Technique for Orthogonalization," J. Inst. Maths Applics, Vol. 5, pp. 162-166, 1969.

[9] Homayoon S.M. Beigi, C. James Li and R.W. Longman, "Learning Control Based on Generalized Secant Methods and Other Numerical Optimization Methods," Sensors, Controls, and Quality Issues in Manufacturing, ASME: Atlanta,PED-Vol.55, pp. 163-175, December 1991.

[10] Andrzej Cichocki and Rolf Unbehauen, "Neural Networks for Solving Systems of Linear Equations and Related Problems," IEEE Transactions on Circuits and Systems, Fundamental Theory and Applications, Vol. 39, No.2, pp. 124-138, February 1992.