A Singular Value Maximizing Data Recording Algorithm for Concurrent Learning

Girish Chowdhary and Eric Johnson

***Abstract*— We present a singular value maximizing algorithm for recording data to be used by concurrent learning adaptive controllers. These controllers use recorded and current data concurrently and can have exponential stability guarantees, with the rate of convergence directly proportional to the minimum singular value of the matrix containing recorded data. The presented algorithm selects data for recording to improve the minimum singular value, and hence results in improved tracking performance, this is established through comparison with previously studied data recording methods that record points that are sufficiently different.**

I. INTRODUCTION

The key capability brought about by concurrent learning adaptive controllers is the ability to guarantee exponential parameter and tracking error convergence without persistency of excitation [4], [3]. Concurrent learning adaptive controllers achieve this by using recorded data concurrently with current data. In our previous work, we showed that convergence can be guaranteed if the recorded data meet a rank-condition [4], [3]. The rank-condition requires that the recorded data contain as many linearly independent elements as the dimension of the basis of the uncertainty. Furthermore, we indicated that the rate of convergence depends on the minimum singular value of the matrix containing the recorded data points (history stack). Therefore, for selecting data for concurrent learning, it would be ideal to record data that meets the rank-condition and maximizes the minimum singular value of the history stack.

If no previous information about a system is available, or changes to the system have rendered the previously available information inapplicable, then a concurrent learning implementation needs to begin with no data points in the memory. In this case, a method for selecting data in real-time is needed, in which instantaneous data will be scanned at regular intervals and data points will be selected for recording if they satisfy selection criteria. In this paper, we discuss several such selection criteria, and present an algorithm that ensures a data point is only included for concurrent learning if the minimum singular value of the history stack matrix is maximized.

The organization of this paper is as follows, we begin with discussion of model reference adaptive control in section II. Concurrent learning adaptive control for adaptive control problems with structured uncertainty is described in section III-A, while neuro-adaptive concurrent learning control for adaptive control problems with unstructured uncertainty is described in section III-B. A simple method for recording different points using a cyclic history stack is described in IV, while the singular value maximizing algorithm is described in section V. The performance of concurrent learning adaptive controllers with the data selection methods is evaluated in section VI. The paper is concluded in section VII.

II. MODEL REFERENCE ADAPTIVE CONTROL

This section discusses the formulation of Model Reference Adaptive Control using approximate model inversion [9], [3]. Let $D_x \in \Re^n$ be compact, and Let $x(t) \in D_x$ be the known state vector, let $\delta \in \Re^k$ denote the control input, and consider the following system:

$$\dot{x} = f(x(t), \delta(t)), \tag{1}$$

where the function f is assumed to be continuously differentiable in $x \in D_x$, and control input δ is assumed to be bounded and piecewise continuous. The conditions for the existence and the uniqueness of the solution to 1 are assumed to be met. Since the exact model 1 is usually not available or not invertible, we introduce an approximate inversion model $\hat{f}(x, \delta)$ which can be inverted to determine the control input δ:

$$\delta = \hat{f}^{-1}(x, \nu). \tag{2}$$

Where ν is the pseudo control input, which represents the desired model output $\dot{x}$ and is expected to be approximately achieved by δ. Hence, the pseudo control input is the output of the approximate inversion model:

$$\nu = \hat{f}(x, \delta). \tag{3}$$

This approximation results in a model error of the form:

$$\dot{x} = \nu(x, \delta) + \Delta(x, \delta) \tag{4}$$

where the model error $\Delta : \Re^{n+k} \to \Re^n$ is given by:

$$\Delta(x, \delta) = f(x, \delta) - \hat{f}(x, \delta). \tag{5}$$

A reference model can be designed that characterizes the desired response of the system:

$$\dot{x}_{rm}(t) = f_{rm}(x_{rm}(t), r(t)), \tag{6}$$

where $f_{rm}(x_{rm}(t), r(t))$ denote the reference model dynamics which are assumed to be continuously differentiable in x for all $x \in D_x \subset \Re^n$. The command $r(t)$ is assumed to be bounded and piecewise continuous, furthermore, it is assumed that all requirements for guaranteeing the existence of a unique solution to 6 are satisfied. It is also assumed that

G. Chowdhary and Assoc. Prof. E. N. Johnson are with the Daniel Guggenheim school of aerospace engineering at the Georgia Institute of Technology, Atlanta GA, Girish.Chowdhary@gatech.edu, Eric.Johnson@ae.gatech.edu

the reference model states remain bounded for a bounded reference input.

A tracking control law consisting of a linear feedback part $u_{pd} = Kx$, a linear feedforward part $u_{crm} = \dot{x}_{rm}$, and an adaptive part $u_{ad}(x)$ is proposed to have the following form:

$$u = u_{crm} + u_{pd} - u_{ad}. \tag{7}$$

Define the tracking error e as $e(t) = x_{rm}(t) - x(t)$, then, letting $A = -K$ the tracking error dynamics are found to be [9], [3]:

$$\dot{e} = Ae + [u_{ad}(x, \delta) - \Delta(x, \delta)]. \tag{8}$$

The baseline full state feedback controller $u_{pd} = Kx$ is assumed to be designed such that A is a Hurwitz matrix. Hence for any positive definite matrix $Q \in \Re^{n \times n}$, a positive definite solution $P \in \Re^{n \times n}$ exists to the Lyapunov equation:

$$A^T P + PA + Q = 0. \tag{9}$$

Letting $\bar{x} = [x, \delta] \in \Re^{n+k}$, two cases for characterizing the uncertainty $\Delta(x)$ can be considered. In the first case (Case I: Structured Uncertainty), the uncertainty is assumed to be represented as an unknown linear combination of known nonlinear basis function. That is, there exist a matrix of constants $W^* \in \Re^{m \times n}$ and a vector of continuously differentiable functions $\Phi(\bar{x}) = [\phi_1(\bar{x}), \phi_2(\bar{x}), ..., \phi_m(\bar{x})]^T$ such that

$$\Delta(\bar{x}) = {W^*}^T \Phi(\bar{x}). \tag{10}$$

In this case letting W denote the estimate W^* the adaptive law can be written as

$$u_{ad}(\bar{x}) = W^T \Phi(\bar{x}). \tag{11}$$

A large class of plants can be modeled in this manner. If the basis functions are also unknown, we say that the uncertainty is *unstructured*, in this case (Case II: Unstructured Uncertainty) if it is known that the uncertainty $\Delta(\bar{x})$ is continuous and defined over a compact domain $D \subset \Re^{n+k}$, a Radial Basis Function (RBF) Neural Network (NN) can be used as the adaptive element:

$$u_{ad}(\bar{x}) = W^T \sigma(\bar{x}). \tag{12}$$

where $W \in \Re^{n \times l}$ and $\sigma(\bar{x}) = [1, \sigma_2(\bar{x}), \sigma_3(\bar{x}), ..., \sigma_l(\bar{x})]^T$ is a vector of known radial basis functions. For $i = 2, 3..., l$ let c_i denote the RBF centroid and μ_i denote the RBF width then for each i The radial basis functions are given as $\sigma_i(x) = e^{-\|\bar{x} - c_i\|^2/\mu_i}$. In this case, the the universal approximation property of RBF NN [14] can be leveraged to guarantee that given a fixed number of radial basis functions l there exists ideal weights $W^* \in \Re^{n \times l}$ and a real number $\tilde{\epsilon}(\bar{x})$ such that $\Delta(x) = {W^*}^T \sigma(\bar{x}) + \tilde{\epsilon}(\bar{x})$.

A. Baseline Adaptive Law

For the case of structured uncertainty it is well known that the following adaptive law

$$\dot{\hat{W}} = -\Gamma_W \Phi(\bar{x}) e^T P \tag{13}$$

where Γ_W is a positive definite matrix of appropriate dimensions results $e(t) \to 0$ [12], [8], [18] if the weights remain bounded. Equation 13 will be referred to as the baseline adaptive law. Furthermore, replacing $\Phi(\bar{x})$ with $\sigma(\bar{x})$ in equation 13 results in the baseline gradient based adaptive law for the case of unstructured uncertainty (case II). For this case, the baseline adaptive law guarantees uniform ultimate boundedness of tracking error e [10].

The baseline adaptive law of equation 13 however, does not guarantee the boundedness of adaptive weights unless modification terms such as σ mod (see [8]), or e mod (see [12]) are added. Furthermore, it is well known that global exponential tracking error ($e(t)$) and weight error ($\tilde{W}(t) = W(t) - W^*$) to zero is only guaranteed for the baseline adaptive law if and only if the plant states (and consequently the signal $\Phi(\bar{x}(t))$) are persistently exciting (PE) [13], [1], [18]. A PE signal must contain as many spectral lines as the dimension of the basis of the uncertainty over a time interval (see reference [18] for a definition of PE signals). Hence, constant reference signals are not PE, nor are exponentially decaying reference signals. In many control applications, enforcing such persistent excitation in the control system may be infeasible or not acceptable due to energy requirements.

III. Concurrent Learning Adaptive Control

Conceptually, the baseline adaptive law attempts to minimize a quadratic cost on the instantaneous tracking error ($\frac{1}{2} e^T(t) e(t)$). This reliance on only instantaneous data results in a rank-‘ update law, that is even though $\dot{W}$ is a matrix, its rank will be at-most one [9], [5]. This is one reason why the baseline adaptive law must be persistently provided with information in order to guarantee exponential stability. A concurrent learning adaptive law on the other hand, uses recorded and current data concurrently for adaptation, and is not rank-1. This ensures that if the recorded data are sufficiently rich, then exponential stability of the zero solution of the tracking error and weight error dynamics can be guaranteed without requiring persistency of excitation. In the following we present some key theorems that characterize these properties.

A. Exponential Convergence with Concurrent Learning for Case of Structured Uncertainty

In this section we present an exponentially stable concurrent learning adaptive controller for the case of structured uncertainty (Case I, in section II) subject to a sufficient verifiable condition on the linear independence of the recorded data.

Theorem 1 Consider the system in equation 1, the control law of equation 7, the case of structured uncertainty. For the j^{th} recorded data point let $\epsilon_j = \nu_{ad}(\bar{x}_j) - \Delta(\bar{x}_j)$, furthermore let p be the number of recorded data points $\Phi(\bar{x}_j)$ in the history stack matrix $Z = [\Phi(\bar{x}_1), ..., \Phi(\bar{x}_p)]$, such that $rank(Z) = m$, and consider the following weight update law:

$$\dot{W}(t) = -\Gamma_W \Phi(\bar{x}(t)) e^T(t) P - \sum_{j=1}^{p} \Gamma_W \Phi(\bar{x}_j) \epsilon_j^T(t). \tag{14}$$

Then the zero solution $e(t) \equiv 0$ of tracking error dynamics of equation 8 is globally exponentially stable and $W(t) \rightarrow W^*$ exponentially. Furthermore, let $\Omega = \sum_{j=1}^{p} \Phi_j \Phi_j^T$, then the rate of convergence is directly proportional to the smallest singular value of Ω.

Proof: A proof can be found in references [3]. An equivalent theorem for a different class of plants is proved in [4]. ■

Remark 1 The above theorem requires only a verifiable condition on the linear independence of the recorded data to guarantee that the zero solutions of the tracking error and the parameter error are globally exponentially stable. It is important to note that the imposed *rank-condition* on the recorded data ($rank(Z) = m$) is significantly different than a condition of persistency of excitation in the states. Firstly, this condition applies only to the recorded data, which is a small subset of all past states, whereas, the persistency of excitation condition applies to all past and future states. Secondly, since the rank of a matrix can be easily determined online, it is possible to verify whether this condition is met online, whereas it is impossible to determine whether a signal will be PE without knowing its future behavior. Hence, the rank-condition required to guarantee convergence when recorded data is concurrently used for adaptation with instantaneous data is less restrictive.

Remark 2 The term $\epsilon_j = \nu_{ad}(\bar{x}_j) - \Delta(\bar{x}_j)$ for the j^{th} data point can be calculated by noting that $\Delta(\bar{x}_j) = \dot{x}_j - \nu(\bar{x}_j)$. Since $\nu(\bar{x}_j)$ is known, the problem of estimating system uncertainty can be reduced to that of estimation of $\dot{x}$. In cases where an explicit measurement for $\dot{x}$ is not available, $\dot{x}_j$ can be estimated using an implementation of a fixed-point smoother [6]. The details of this process are described in [3], [5].

B. Neuro-Adaptive Control with Guaranteed Boundedness with Concurrent Learning for case of Unstructured Uncertainty

When the structure of the uncertainty is unknown (case II in section II), a RBF NN can be used as the adaptive element by leveraging their universal approximation property of RBF NN. In contrast with the baseline adaptive law $\dot{W} = -\Gamma_W \sigma(\bar{x}) e^T P$, a concurrent learning adaptive law guarantees that the tracking error approaches and remains bounded in a neighorhood of zero, and the weight error approaches and remains bounded in a compact neighborhood of the ideas weights. This is proven in reference [3].

IV. A Simple Method for Recording Sufficiently Different Points

In the following, we assume that the recorded data is stored in a history-stack, and new data is added or old data removed based on the criteria discussed below. We will let $p \in \aleph$ denote the subscript of the last point stored. For ease of exposition, for a stored data point x_j, we let $\Phi_j \in \Re^m$ denote $\Phi(x_j)$ the data point to be stored. We will let $Z_k = [\Phi_1, ..., \Phi_p]$ denote the history stack at time step k. The p^{th} column of Z_k will be denoted by $Z_k(:, p)$. It is assumed that the maximum allowable number of recorded data points is limited due to memory or processing power considerations. Therefore, we will require that Z_k has a maximum of $\bar{p} \in \aleph$ columns, clearly, in order to be able to satisfy the rank-condition (see remark 1), $\bar{p} \geq m$. For the j^{th} data point, the associated model error $\Delta(x_j)$ is assumed to be stored in the array $\bar{\Delta}(:, j) = \Delta(x_j)$.

For a given $\epsilon \in \Re^+$ a simple way select the instantaneous data $\Phi(x(t))$ for recording is to require:

$$\frac{\|\Phi(x(t)) - \Phi_p\|^2}{\|\Phi(x(t))\|} \geq \epsilon. \tag{15}$$

The above method ascertains that only those data points are selected for storage that are sufficiently different from the last data point stored. In order to meet the dimension of the history stack, the data can be stored in a cyclic manner. That is if $p = \bar{p}$, then the next data point replaces the oldest data point (Φ_1), and so on. This method has been used for selecting data points in our previous work [5], [4], and was found to be effective.

If the mapping Φ has the properties of a logistic function (see for example [7]) then it is sufficient to pick sufficiently different x_k in order to achieve the same effect as that of equation 15. This property is useful when dealing with Neural Network (NN) based adaptive controllers, particularly since in these cases the dimension of Φ is often greater than the dimension of x. Furthermore, due to Micchelli's theorem, the satisfaction of the rank-condition for Radial Basis Function NN is reduced to selecting distinct points for storage [11], [7]. Hence in this particular case, the criterion in equation 15 is an effective and efficient way of selecting data points for recording. However, for general cases, this method does not guarantee that the rank-condition will always be satisfied.

V. A Singular Value Maximizing Approach

From theorem 1 we have that the rate of convergence depends on $\lambda_{\min}(\Omega)$. Letting $\sigma(\Omega)$ denote the singular values of Ω, we recall that for nonzero singular values $\sigma(\Omega) = \sqrt{\lambda(\Omega\Omega^T)}$, and Ω is full ranked only if $\sigma_{\min}(\Omega)$ is nonzero [17]. This fact can be used to select data points for storage. The method we present in this section selects a data point for recording if its inclusion results in an increase in the instantaneous minimum singular value of Ω. The following fact ascertains that the singular values of Ω are the same as that of Z_k.

Fact 2 $\sigma_{\min}([\Phi_1, ..., \Phi_p]) = \sqrt{\sigma_{\min}(\sum_{j=1}^{p} \Phi_j \Phi_j^T)}$

Proof:

As before, let $Z_k = [\Phi_1, ..., \Phi_p]$, then $\sum_{j=1}^{p} \Phi_j \Phi_j^T = [\Phi_1, ..., \Phi_p][\Phi_1, ..., \Phi_p]^T = Z_k Z_k^T$. The proof now follows by noting that $\sigma_{\min}(Z_k) = \sqrt{\lambda_{\min}(Z_k Z_k^T)} = \sqrt{\sigma_{\min}(Z_k Z_k^T)}$.

The proof now follows by noting that . ■

Leveraging this fact, algorithm 1 aims to maximize the minimum singular value of the matrix containing the history stack. The algorithm begins by using criterion in equation 15 to select sufficiently different points for storage. If the number of stored points increases the maximum allowable number, the algorithm seeks to incorporate new data points in such a way that the minimum singular value of Z_k is increased. To achieve this, the algorithm sequentially replaces every recorded data point in the history stack with the current data point and stores the resulting minimum singular value in a variable. The algorithm then finds the maximum over these values, and accepts the new data point for storage into the history stack (by replacing the corresponding existing point) if the resulting configuration results in an increase in the instantaneous minimum singular value of Ω.

Algorithm 1 Singular Value Maximizing Algorithm for Recording Data Points

```
Require: $p \geq 1$
  if $\frac{\|\Phi(x(t)) - \Phi_p\|^2}{\|\Phi(x(t))\|} \geq \epsilon$ then
    $p = p + 1$
    $Z_k(:, p) = \Phi(x(t))$; {store $\bar{\Delta}(:, p) = \Delta(x(t))$}
  end if
  if $p \geq \bar{p}$ then
    $T = Z_k$
    $S_{old} = \min SVD(Z_k^T)$
    for $j = 1$ to $p$ do
      $Z_k(:, j) = \Phi(x(t))$
      $S(j) = \min SVD(Z_k^T)$
      $Z_k = T$
    end for
    find $\max S$ and let $k$ denote the corresponding column index
    if $\max S > S_{old}$ then
      $Z_k(:, k) = \Phi(x(t))$, {store $\bar{\Delta}(:, k) = \Delta(x(t))$}
      $p = p - 1$
    else
      $p = p - 1$
      $Z_k = T$
    end if
  end if
```

VI. Evaluation of Data Point Selection Methods Through Simulation

In this section we evaluate the effectiveness of the data point selection criteria through numerical simulation on a wing rock dynamics model. Wing rock is caused due to asymmetric stalling on lifting surfaces of agile aircraft. If left uncontrolled, the oscillations caused by wing rock can easily grow unbounded and cause structural damage [15]. Let ϕ denote the roll angle of an aircraft, p denote the roll rate, δ_a denote the aileron control input, then a simplified model for wing rock dynamics is given by:

$$\dot{\phi} = p \tag{16}$$

$$\dot{p} = \delta_a + \Delta(x). \tag{17}$$

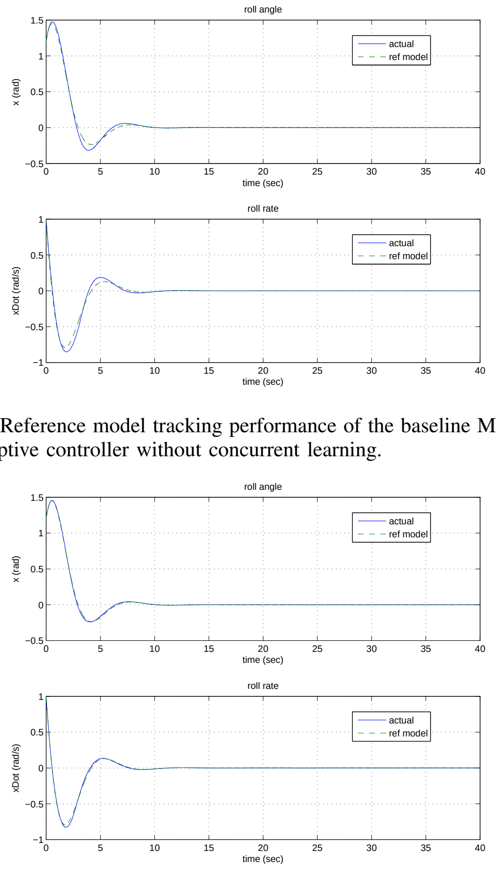
Where $\Delta(x) = W_0 + W_1\phi + W_2 p + W_3|\phi|p + W_4|p|p + W_5\phi^3$. The parameters for wing rock motion are adapted from [16], they are $W_0 = 0.0, W_1 = 0.2314, W_2 = 0.6918, W_3 = -0.6245, W_4 = 0.0095, W_5 = 0.0214$. Initial conditions for the simulation are arbitrarily chosen to be $\phi = 1.2deg, p = 1deg/s$. An MRAC controller (see section II) is used. The reference model chosen is a stable second order linear system with natural frequency of 1 $radian/second$ and damping ratio of 0.5. The linear control gains are given by $K = [2.5, 2.3]$, and the learning rate is set to $\Gamma_W = 2$. The simulation runs for a total time of 40 seconds with an update rate of 0.005 seconds using Euler integration. The reference model tracking performance of the baseline MRAC algorithm (without concurrent learning) is shown in 1(a), while the reference model tracking performance of the concurrent learning MRAC adaptive controller with singular value maximizing data point selection (algorithm 1) is shown in figure 1(b). For the chosen learning rate, we note that the concurrent learning adaptive controller is better at tracking the reference model. In this simulation however, we are concerned more with the impact of the selection of data points on weight convergence. To that effect, we will evaluate the different data point selection criterion separately in the following.

A. Weight Evolution without Concurrent Learning

Figure 2 shows the evolution of weights when using the baseline MRAC controller without concurrent learning. We note that the weights do not converge to their ideal values. Furthermore, once the states arrive at the origin (that is once $\phi = 0,\ p = 0$) the weights are no longer updated. This is expected in a controller that only uses instantaneous data for adaptation.

B. Weight Evolution with a Static History Stack

For the results presented in this section, we use a static history stack with a fixed number of slots. The history stack here is called static because once a data point is recorded, it permanently occupies a slot in the history stack and cannot overwritten. The data points are selected using the criterion in equation 15 with $\epsilon = 0.08$. Figure 3 shows the evolution of the weights for a simulation run. It is interesting to note that the weights continue to be updated even after the states arrive at the origin. This is an effect of concurrent training on recorded data. In fact, it can be seen that for the chosen learning rate and the data point selection criterion, the weights are approaching their true values, however are not sufficiently close to the ideal values by the end of the simulation. At the end of the simulation it was found that $\sigma_{\min}(\Omega) = 0.0265$



(a) Reference model tracking performance of the baseline MRAC adaptive controller without concurrent learning.

(b) Reference model tracking performance of the concurrent learning adaptive controller with singular value maximizing data point selection (see algorithm 1).

Fig. 1. Comparison of reference model tracing performance for the control of wing rock dynamics with and without concurrent learning.

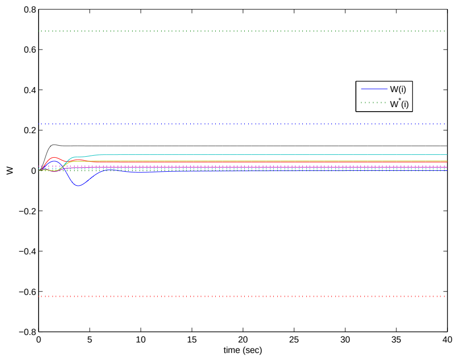


Fig. 2. Evolution of weight when using the baseline MRAC controller without concurrent learning. Note that the weights do not converge, in fact, once the states arrive at the origin weights remain constant.

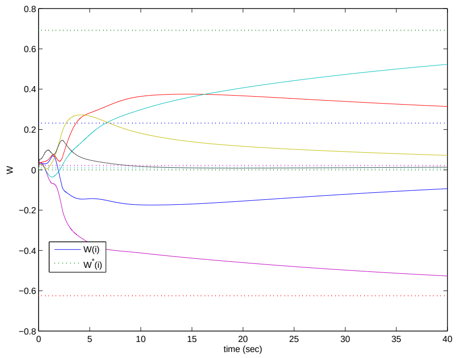


Fig. 3. Evolution of weight with concurrent learning adaptive controller using a static history stack. Note that the weights are approaching their true values, however are not close to the ideal value by the end of the simulation (40 seconds).

C. Weight Evolution with a Cyclic History Stack

The history stack here is called cyclic because data is recorded in a cyclical manner. That is, once the history stack is full, the newest data point bumps out the oldest data point and so on. This approach aid in guaranteeing that the history stack reflects the most recently stored data points. The data points are selected using the criterion in equation 15 with $\epsilon = 0.08$. Figure 4 shows the evolution of the weights for a simulation run. As in the previous case, concurrent learning results in weight update even after the states arrive at the origin. It can be seen that the weights are closer to their true values than when using a static history stack. At the end of the simulation it was found that $\sigma_{\min}(\Omega) = 0.0980$.

D. Weight Evolution with the Singular Value Maximizing Approach

In this simulation run, the data points are recorded using algorithm 1. Figure 5 shows that the weights converge to their true values within 20 seconds of the simulation. Furthermore, convergence occurs even when the states have arrived at the origin and are no longer PE. At the end of the simulation it was found that $\sigma_{\min}(\Omega) = 0.3519$. Figure 6 compares $\sigma_{\min}(\Omega)$ at every time step for the three data point selection algorithms discussed. It can be seen that when using a static history stack, $\sigma_{\min}(\Omega)$ reaches a constant value and remains there once the history stack is full. Whereas, when a cyclic history stack is used, $\sigma_{\min}(\Omega)$ changes as new data replaces old data and occasionally even drops below $\sigma_{\min}(\Omega)$ achieved when using a static history stack, however by the end of the simulation $\sigma_{\min}(\Omega)$ with a cyclic history stack is larger than $\sigma_{\min}(\Omega)$ when using a static history stack. The singular value maximizing algorithm (algorithm 1) outperforms both these methods. It can be seen that new data points are selected and old data points removed such that the minimum singular value is maximized. This improvement in the quality of the data is also reflected in weight convergence, with the weights updated by the singular value maximizing approach arriving at their true values faster than the other two approaches.

E. Concurrent Learning with Pre-recorded history stack

If a pre-recorded history stack satisfying the rank-condition is available for a system, then exponential stability of the tracking error and weight error dynamics can be guaranteed. These results are presented in [2]. Note that the

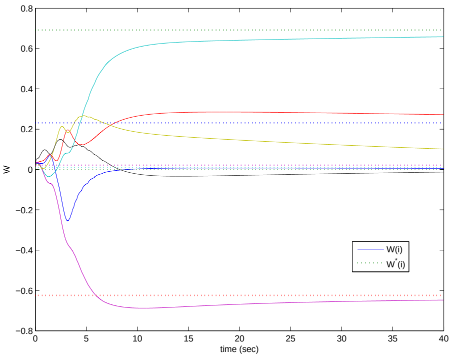


Fig. 4. Evolution of weight with concurrent learning adaptive controller using a cyclic history stack. Note that the weights are approaching their true values, and they are closer to their true values than when using a static history stack within the first 20 seconds of the simulation.

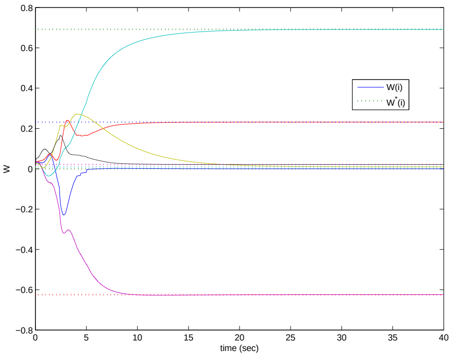


Fig. 5. Evolution of weight with concurrent learning adaptive controller using the singular value maximizing algorithm (algorithm 1). Note that the weights approach their true values by the end of the simulation (40 seconds).

exponential stability can be used to formulate exponentially decaying transient performance bounds.

VII. CONCLUSION

We presented a singular value maximizing algorithm for recording data points for concurrent learning. A simulation of wing rock dynamics was used to characterize the effectiveness of the approach. We compared the performance of this approach with previously studied approaches, including a cyclic history stack approach in which newer data points bump out older data points. We conclude that the singular value maximizing algorithm presented in this paper ensures a rich stack of recorded data is maintained while staying within memory limits. Future work includes algorithms to monitor the relevance of recorded data to current operating conditions to determine if history stack repopulation is required.

VIII. ACKNOWLEDGMENTS

This work was supported in part by NSF ECS-0238993 and NASA Cooperative Agreement NNX08AD06A.

REFERENCES

[1] Stephan Boyd and Sankar Sastry. Necessary and sufficient conditions for parameter convergence in adaptive control. *Automatica*, 22(6):629–639, 1986.

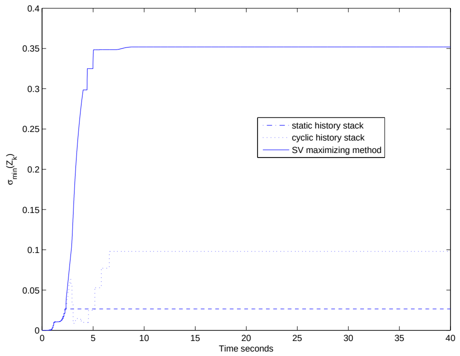


Fig. 6. Plot of the minimum singular value $\sigma_{\min}(\Omega)$ at every time step for the three data point selection criteria discussed. Note that in case of the static history stack, $\sigma_{\min}(\Omega)$ stays constant once the history stack is full, in case of the cyclic history stack, $\sigma_{\min}(\Omega)$ changes with time as new data replace old data, occasionally dropping below that of the $\sigma_{\min}(\Omega)$ for the static history stack. When the singular value maximizing algorithm (algorithm 1) is used, data points are only selected such that $\sigma_{\min}(\Omega)$ increases with time. This results in faster weight convergence.

[2] G. Chowdhary and E. N. Johnson. Adaptive flight control with guaranteed convergence. In *Conference on Guidance, Navigation and Control*, Munich, Germany, April 2011. CEAS.

[3] Girish Chowdhary. *Concurrent Learning for Convergence in Adaptive Control Without Persistency of Excitation*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 2010.

[4] Girish Chowdhary and Eric N. Johnson. Concurrent learning for convergence in adaptive control without persistency of excitation. In *49th IEEE Conference on Decision and Control*, 2010.

[5] Girish Chowdhary and Eric N. Johnson. Theory and flight test validation of a concurrent learning adaptive controller. *Journal of Guidance Control and Dynamics*, 34(2):592–607, March 2011.

[6] Arthur Gelb. *Applied Optimal Estimation*. MIT Press, Cambridge, 1974.

[7] Simon Haykin. *Neural Networks a Comprehensive Foundation*. Prentice Hall, USA, Upper Saddle River, 2 edition, 1998.

[8] Petros A. Ioannou and Jung Sun. *Robust Adaptive Control*. Prentice-Hall, Upper Saddle River, 1996.

[9] Eric N. Johnson. *Limited Authority Adaptive Flight Control*. PhD thesis, Georgia Institute of Technology, Atlanta Ga, 2000.

[10] Y. H. Kim and F.L. Lewis. *High-Level Feedback Control with Neural Networks*, volume 21 of *Robotics and Intelligent Systems*. World Scientific, Singapore, 1998.

[11] Charles A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Construct. Approx.*, 2(1):11 –22, dec. 1986.

[12] K. Narendra and A. Annaswamy. A new adaptive law for robust adaptation without persistent excitation. *IEEE Transactions on Automatic Control*, 32(2):134–145, February 1987.

[13] Kumpati S. Narendra and Anuradha M. Annaswamy. *Stable Adaptive Systems*. Prentice-Hall, Englewood Cliffs, 1989.

[14] J. Park and I.W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computatations*, 3:246–257, 1991.

[15] Ahmed A. Saad. *Simulation and analysis of wing rock physics for a generic fighter model with three degrees of freedom*. PhD thesis, Air Force Institute of Technology, Air University, Wright-Patterson Air Force Base, Dayton, Ohio, 2000.

[16] S. N. Singh, W. Yim, and W. R. Wells. Direct adaptive control of wing rock motion of slender delta wings. *Journal of Guidance Control and Dynamics*, 18(1):25–30, Feb. 1995.

[17] Gilbert Strang. *Linear Algebra and its Applications*. Thomson Learning, Brooks, 1988.

[18] Gang Tao. *Adaptive Control Design and Analysis*. Wiley, New York, 2003.