ARI The Bulletin of the Istanbul Technical University VOLUME 53, NUMBER 1
Communicated by Sondan Durukanoğlu Feyiz

Monitoring Jet Engine Characteristics by a Learning Algorithm

Nafiz Aydın Hızal
Department of Mechanical Engineering, Istanbul Technical University, 80191 Gümüşsuyu, Istanbul, Turkey
(Received 10 July 2002)

Characteristic curves or surfaces of engines are important regarding the safety and the efficiency of operation. Deriving these functions automatically, and then monitoring them constantly against their longer term variations would provide the necessary information required to find the optimal operating points, to detect impending failures or to diagnose after a failure. This paper discusses the use of a learning algorithm for this purpose. The algorithm has originally been used for building heating automation, and in this paper the feasibility and the effects of the algorithm parameters for a jet engine fuel consumption application are investigated after presenting the algorithm and studying some of its fundamental properties. The engine characteristic used in simulations is that of a typical turbofan aircraft engine, and its Thrust-Specific-Fuel-Consumption is derived against the Mach number and the thrust as the independent variables, serving as an example and also illustrating some further properties of the algorithm.

Keywords: Learning systems, engine characteristics, jet engine characteristics, optimal operation of engines.

1. Introduction

Characteristics of internal combustion engines provide an indication of their conditions as to whether or not they are operating normally or beyond acceptable limits of performance, regarding the power or thrust output, the fuel consumption (efficiency) and the pollution generating potential. For this reason, deriving and constantly monitoring these characteristics automatically would prove useful in the efforts to make the engine run at its best operating point with regard to power, efficiency or pollution in spite of possible long term variations, in addition to its role as an early warning system against impending trouble, and perhaps as an aid in diagnosis after a failure.

In this study, a trainable nonlinear function generator is proposed for this purpose. The paper first presents the algorithm and studies some additional properties of it, then, as an example usage, considers the application of the algorithm to the monitoring of the "Thrust Specific Fuel Consumption" surface of a typical turbine engine, where the effects of the parameters of this estimation and learning algorithm are investigated more specifically, based on simulations.

As described in the following section, the algorithm is one that has been tested successfully in practical commercial applications concerned with building heating automation. It is not derived from a mathematical starting point such as aiming for the minimization of the squared error, because factors other than mathematical criteria usually have to be taken into account in the definition of such practically feasible algorithms to obtain efficiency, reliability and simplicity. Apparently, this practically proven algorithm has the potential to be useful in many engineering applications, and this potential has motivated this study to understand its properties better, through its application on a jet engine characteristic surface monitoring task.

2. The Structure of the Algorithm

The algorithm proposed here for deriving and monitoring engine characteristic curves, surfaces, or hypersurfaces has been proposed and used for building heating automation, concerned with heating efficiency in particular (Leimgruber et al. 1984, 1988). For buildings like business headquarters, malls, or schools which are not heated 24 hours a day, the system estimates the best turn-on and turn-off times for maximizing the efficiency of the heating system in this regard, without compromising comfort, by learning the building dynamics related to this function. In this application, a two dimensional input space involving the building and the ambient temperatures is used. It has been shown that by making the algorithm three dimensional and adding the

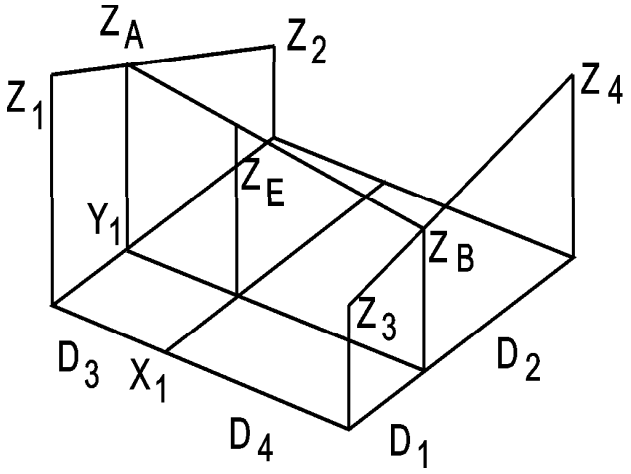

Figure 1. One cell of the input space. Distances of the input point (X_1,Y_1) from the edges are Di.

wind speed as a third input variable, the performance is improved especially for windy regions (Hızal 1997). The use of this learning algorithm in a gain scheduling adaptive control scheme has also been studied (Hızal 1999).

The algorithm has the estimation and the training stages. Estimation is performed by multidimensional linear interpolation, using a number of "support" values. Training is based on the correction of these support values according to the estimation error and to the degree of their contributions to the estimation process.

In the original two dimensional form of the algorithm, the ranges of the input variables X and Y are subdivided into m and n regions, respectively, thus generating an (m+1) by (n+1) dimensioned grid, at the intersections of which the support values will be located. The estimation at a particular input point (X_1,Y_1) gives the estimate $Z_E = f_E(X_1,Y_1)$. The actual Z value Z_M must be measured at the time of the estimation (as in the engine characteristics application) or after using the estimate (as in the building automation application) so that the error can be used in the training stage following this estimation.

A single cell of the input space grid is shown in Figure 1. For the estimation stage, first it has to be determined in which cell the input point (X_1,Y_1) is, since the support values to be used are the ones that are at the corners of this particular cell, namely, Z_i (i =1,...,4). The distances of the point (X_1,Y_1) to the corners of the cell are D_i (i = 1,...,4). Two linear interpolations in the same direction are carried out at the two opposed edges of the cell, resulting in the auxiliary values Z_A and Z_B which are used for the third interpolation in the other direction, completing a bilinear interpolation giving the estimate Z_E. This procedure at a particular time step k is represented by the following equations where the time index k is omitted for the sake of clarity.

$$Z_A = Z_1 + (Z_2 - Z_1)D_1/(D_1 + D_2) \quad (1)$$

$$Z_B = Z_3 + (Z_4 - Z_3)D_1/(D_1 + D_2) \quad (2)$$

$$Z_E = Z_A + (Z_B - Z_A)D_3/(D_3 + D_4) \quad (3)$$

For the training stage, the actual value Z_M is used to calculate the "error factor" F which is the relative error as defined below.

$$F = (Z_M - Z_E)/Z_E \quad (4)$$

The four support values Z_i are corrected according to the error factor F and by using weights K_i that represent their contributions to the estimate. The weights K_i are calculated as follows, as functions of the distances D_i.

$$K_1 = [D_2/(D_1 + D_2)][D_4/(D_3 + D_4)] \quad (5)$$

$$K_2 = [D_1/(D_1 + D_2)][D_4/(D_3 + D_4)] \quad (6)$$

$$K_3 = [D_2/(D_1 + D_2)][D_3/(D_3 + D_4)] \quad (7)$$

$$K_4 = [D_1/(D_1 + D_2)][D_3/(D_3 + D_4)] \quad (8)$$

Equation (9) below gives the correction expression for the i'th support value, its new value Z_i(k+1) being calculated from the quantities at the time step k.

$$Z_i(k+1) = Z_i(k)[1 + K_i(k)F(k)] \quad (9)$$

Some attributes of this nonlinear algorithm is investigated in (Hızal 1998), like the investigation of the effects of the addition of a "convergence constant" or "forgetting factor" μ to the algorithm (Åström and Wittenmark 1989), so that the training rule (9) becomes

$$Z_i(k+1) = Z_i(k)[1 + \mu K_i(k)F(k)] \quad (10)$$

and the investigation of the effects of the cell size.

As pointed out in (Hızal 1998), repeated training at one point constitutes a simplified case, allowing a linear analysis. If this point of repeated training is taken as a support point, then a stability bound can be obtained as given by the following theorem.

Theorem: For the learning algorithm defined by Eqs.(1-8, 10), the convergence constant value $\mu = 2$ constitutes a lower bound for the range of μ values that can result in unstable operation, under any distribution of the training points over the input space.

Proof: Consider a point (X_1,Y_1) in the input space such that the trained support values (those with nonzero weighting constants) Z_i have equal weighting constants K due to symmetry. Starting from equal initial support values, all $Z_i = Z$ will satisfy the same linear difference equation

$$Z(k+1) = Z(k)[1+\mu K(Z_M(k)-Z(k))/Z(k)] \quad (11)$$

which can be obtained from Eqs.(1-8, 10) under the stated conditions, with zero initial values. Taking the z-transform of (11), the z-transfer function G(z) = Z(z) / Z_M(z) can be written as

$$G(z) = \mu K/(z-(1-\mu K)) \quad (12)$$

whose pole is at p = 1 - μ K. As the pole magnitude must be smaller than unity for stable operation (Åström and Wittenmark 1984), the following limits are found for μ to preserve stability.

$$|1-\mu K| < 1 \quad or \quad 0 < \mu K < 2 \quad (13)$$

The worst case regarding stability, among all the operating conditions, is when a support point is trained repeatedly (when the input point repeatedly coincides with a support point), because then the weight K for the trained point is at its maximum value of unity, forcing μ to have the minimum value for stability. Therefore, as a result of (13), $\mu < 2$ leads to stable operation under any distribution of the training points over the input space.

3. Jet Engine Characteristics

Since reliability and efficiency issues are of utmost importance in jet engines, many variables can be considered as the dependent variable to be monitored by the learning algorithm in a jet engine. Two outstanding candidates are the thrust and the fuel consumption.

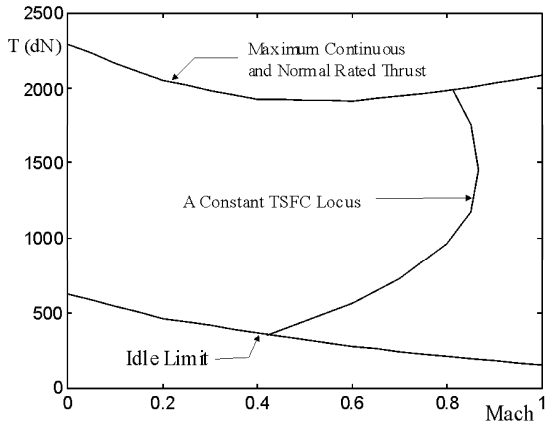

Figure 2. The operating region of the JT3D-1 engine in the Mach number - Thrust plane.

For any one dependent variable to be monitored, a large number of independent input variables exist. However, due to a result of the π theorem which states that any function of n variables involving q fundamental quantities such as length, mass, temperature, time, can be reduced to a function of (n - q) dimensionless groups (Hill and Peterson 1967), these relationships can be substantially simplified, making their realization with a small input space dimensionality feasible.

Generally, the most important variables for the characteristics of a particular jet engine are the Mach number, the ambient pressure and temperature, the shaft speed, the thrust, the thrust specific fuel consumption, and the air flow rate (Hill and Peterson 1967, Mattingly et al. 1987, Kerrebrock 1992, Treager 1996) .

The thrust T for a particular turbine engine can be expressed as a function of the Mach number M, the shaft speed N, the ambient-to-standard temperature ratio θ, the ambient-to-standard pressure ratio δ, as (Hill and Peterson 1967)

$$\frac{T}{\delta} = f\left(M, \frac{N}{\sqrt{\theta}}\right) \quad (14)$$

T / δ and $N/\sqrt{\theta}$ are the "corrected" thrust and shaft speed respectively.

As seen from this expression, the surface representing the thrust variable can be formed as a function of only two inputs by an algorithm with

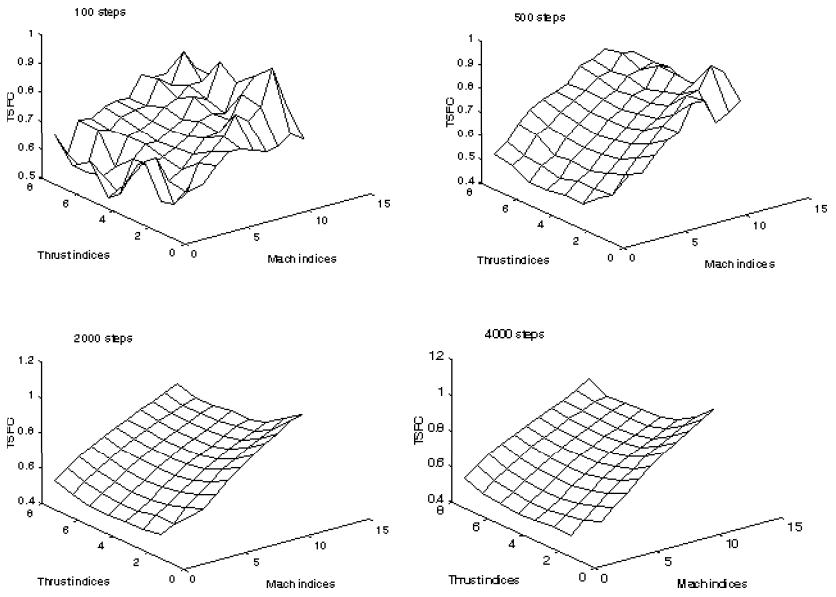

Figure 3. Development of the 3 dimensional TSFC surface as a result of learning, under the conditions $D_S = 2$ and $\mu = 1$.

a two dimensional input space, for a wide range of operating conditions.

The "Thrust Specific Fuel Consumption" (TSFC) for a jet engine is defined as the fuel flow rate per unit thrust.

$$TSFC = (dm_f/dt)/T \tag{15}$$

where m_f is the amount of fuel used. In this case, altitude has a direct effect through its effects on the combustor efficiency. As the effect of the altitude on combustor efficiency is quite complicated, its effect on TSFC is not as clear as that on thrust. If the effects of the combustor efficiency η_c could be ignored, TSFC would be written as a function of M and $N/\sqrt{\theta}$ similar to (14), but actually the product η_c(TSFC) can be written as in (14), and the altitude has to enter the expression as a measured variable to give η_c through experimental relations (Hill and Peterson 1967).

The characteristics used in the simulation are those of JT3D-1, a typical turbofan engine by Pratt & Whitney Aircraft, United Aircraft Corporation, at an altitude of 10700 meters, given as estimated two dimensional curves in (Kerrebrock 1992).

4. Application of The Learning Algorithm

A reference matrix $\mathbf{M}_R$ is produced from the two dimensional characteristic curves of TSFC as a function of M and T, by a routine that finds the reference TSFC values at any point within the M and T ranges considered, by bicubic interpolation. The Mach number range considered is (0,1) and the thrust range is (445,2225) dekaNewtons. Within these ranges, some parts are out of the operating region due to the idle limit and the maximum power limit (Figure 2). During the generation of the input points (M,T) randomly, a Mach number value is selected in the range (0,1) first, and then a thrust value is selected observing the upper and lower thrust limits for the particular Mach number selected.

The support values are also stored as a matrix, $\mathbf{M}_S$. Three cell sizes ("support matrix densities" as referred to here) for the support matrix $\mathbf{M}_S$ are used for comparison of the cell size effects. These densities are called D_S= 1, 2, and 3 corresponding to the support matrix dimensions 6x5, 11x9, and 21x17, respectively, all covering the same M and T ranges given above, resulting in 20, 80, and

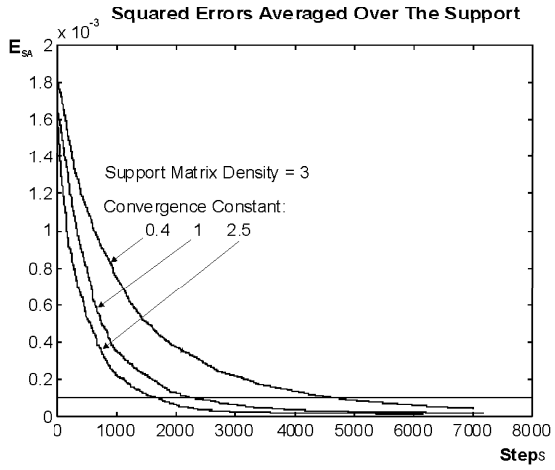

Figure 4. Variation of E_{SA} in time, for $D_S = 3$ and the three μ values.

320 as the numbers of cells, halving the linear cell dimensions with each density increase.

For the convergence constant μ, the three values 0.4, 1, and 2.5 are used, with $\mu = 1$ corresponding to the original form of the algorithm.

The main points of concern are the convergence speed, the steady-state accuracy, and the steady-state error variances. These are investigated as functions of the support matrix density D_S and the convergence constant μ.

5. Simulation Results

The development of the three dimensional "estimated TSFC" surface in time for the conditions $D_S = 2$ and $\mu = 1$ and uniform initial support values of 0.7 is given in Figure 3.

The convergence speed is best evaluated by observing a learning curve produced as the variation of the average squared error E_{SA} in time. The average is over all the support values at a given time step k, i.e.,

$$E_{SA}(k) = [\Sigma_{i,j}(\mathbf{M}_R(i,j) - \mathbf{M}_{Sk}(i,j))^2]/n_S \quad (16)$$

where $\mathbf{M}_{Sk}$ stands for the support matrix at time step k, and n_S is the number of support values. The learning process is started by making the initial support matrix $\mathbf{M}_S(0)$ equal to the reference matrix $\mathbf{M}_R$ with $\pm(0$ - $10)\%$ uniformly distributed random errors added, though this did not make great differences in convergence times in most cases, with a criterion such as $E_{SA} = 10^{-4}$ because when a uniform initial matrix is used, the level of accuracy corresponding to the above $\mathbf{M}_S(0)$ (on the order of $2\text{x}10^{-3}$) is reached quite fast compared to the number of steps required to reach the given E_{SA}. Figure 4 shows the variation of E_{SA} in time, with the three μ values, and with $D_S = 3$. Figure 5 shows the same variable with the three D_S values, with $\mu = 0.4$. As expected, larger convergence constant μ values result in greater convergence speeds. On the other hand, greater support value densities decrease the speed, but not in proportion with the number of support values. This is interpreted as the effects of the cells at the edges and corners. A support point that is away from the edges is trained by inputs within the 4 cells that the support point belongs to. This number is 2 for a support at an edge, and 1 for a support at a corner. For a uniformly distributed (X_1,Y_1), these supports are trained with average relative frequencies of 1, 0.5 and 0.25, respectively, affecting their individual rates of convergence. When the number of cells is less, this effect is more pronounced. Compared to an infinite sized support matrix, an algorithm with a finite number of cells mxn would have its speed reduced by the factor α given by

$$\begin{aligned} \alpha &= [(m-1)(n-1) + 0.5(2(m-1) \quad (17) \\ &+ 2(n-1)) + 0.25(4)]/[(m+1)(n+1)] \end{aligned}$$

which is the relative speeds weighted by the proportions of the numbers of supports for the three categories. This phenomenon is observed at the earlier stages of the learning process, e.g., when E_{SA} halving times are compared, and also as the apparent correction of the supports at the edges taking longer time while the support matrix is being watched graphically during a run. At the later stages the smaller error potential of greater support matrix densities outweighs this effect and it is possible for a denser support matrix to settle in a smaller number of steps, as shown in Figure 6. In this figure the convergence measure is taken as the average number of steps to reach the level $E_{SA} = 10^{-4}$ which can be attained by all the support matrix densities used.

During several trials for stability, sporadic instability was observed for $\mu > 3.5$ with all support matrix densities used. However, during the large number of runs with $\mu \leq 2.5$, no instability was encountered.

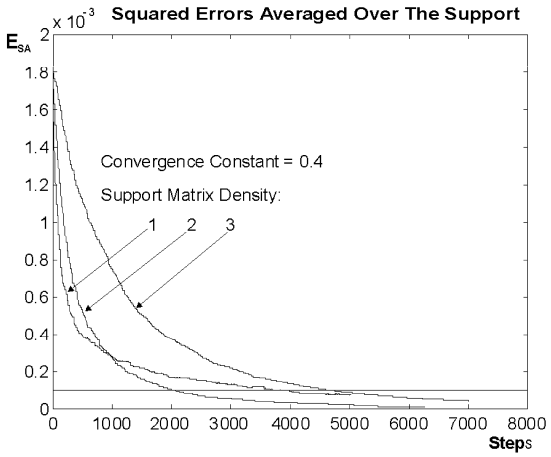

Figure 5. Variation of E_{SA} in time, for $\mu = 0.4$ and the three D_S values.

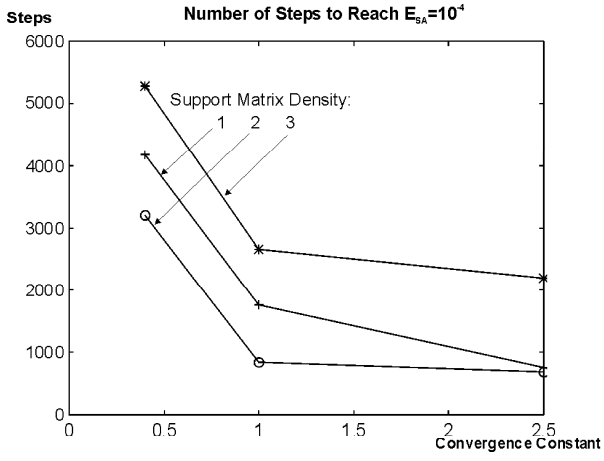

Figure 6. Average number of steps required to reach $E_{SA} = 10^{-4}$ under the nine operating conditions investigated.

The E_{SA} values used for convergence speed evaluation reflect a bias in the steady state, because the average squared errors are evaluated at support points only. Because of the curvature of the generated surface and the finite cell size, the algorithm has to make the support values lower than the actual values at these points, under the effect of the points towards the centers of the cells. When the errors are evaluated over the entire input space, the average is practically zero, so the estimates are unbiased in general, though they are locally biased. These local biases are reduced by using greater support matrix densities, i.e., smaller cells.

Also because of these biases, local adaptations conflict with each other, causing support value variations in the steady state. This effect is more pronounced with a larger μ value, so that small μ values result in smoother operation in the steady state. The error variances reflect this effect. The support value matrix density has much greater effect in this respect, however.

Small μ values are favorable also for the case with measurement noise in the input variables, because of the more pronounced low pass filtering effect resulting from such values. Indeed, considering the discrete transfer function (12) for repetitive training at one point, the amplitude ratio from the corresponding frequency response function is (Leigh 1985)

$$|G(j\omega)| = \mu K/[(cos(\omega T_S) - (1 - \mu K))^2 + (sin(\omega T_S))^2]^{1/2} \quad (18)$$

where T_S is the sampling period. This function exhibits low-pass character if $\mu\ K < 1$. Its breakpoint frequency at -3 decibel amplitude ratio can be found by $|G(j\omega_b)| = (1/2)^{1/2}$ as

$$\omega_b = cos^{-1}[1 + (\mu K)^2/(2\mu K - 2)]/T_S$$
$$\mu K < 0.8284 \quad (19)$$

and its minimum amplitude ratio which is at the Nyquist frequency of $\omega_N = \pi\ /\ T_S$ can be found by substitution of ω_N in (18), as

$$|G|_{min} = \mu K/(2 - \mu K) \quad (20)$$

Both of these functions are monotonically increasing functions of the convergence constant μ,

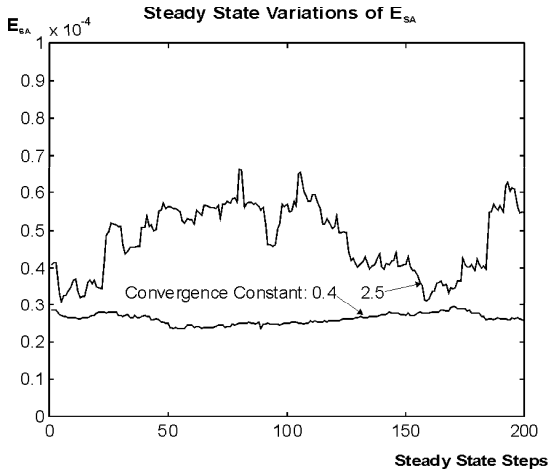

Figure 7. Typical steady state responses in terms of E_{SA}, given for D_S =1 and with the two μ values 0.4 and 2.5, to depict the role of μ in steady state.

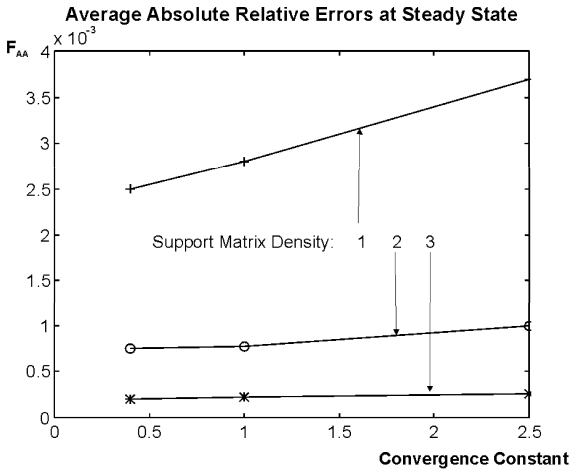

Figure 8. F_{AA}, the average absolute value of the error factor F, as a function of D_S and μ.

indicating the greater low-pass effect of a smaller μ value.

Figure 7 compares typical steady state E_{SA} responses for $D_S = 1$, with $\mu = 0.4$ and $\mu = 2.5$. Steady state time averages of the absolute values of the error factors F, which is denoted by F_{AA} are given in Figure 8, and their variances are given in Figure 9. As pointed out in (Hızal 1998), the steady state errors decrease approximately as the square of the cell size, as can be seen from Figure 8. This is because of the geometry in Figure 10, which assumes circular variations of the actual values in a vertical plane, within distances comparable to cell sizes. In this case, the maximum error h can be given as an approximate function of the radius of curvature r and the support value distance Δ as (Hızal 1998)

$$h \cong \Delta^2/(8r) \tag{21}$$

The variances of F_{AA}, on the other hand, can be seen to change by about an order of magnitude with a decrease in cell size by a factor of 1/2.

For steady state investigations, "reverse training" can be used in simulations to reach steady state conditions in a much smaller number of steps. Starting with $\mathbf{M}_S = \mathbf{M}_R$ with the μ value whose steady state effect is to be investigated, the steady state error conditions are reached from below.

Usual error analyses involving algorithms whose convergence constants tend to zero in time are not valid in this case, for the following two reasons. This algorithm has to have a nonzero convergence constant μ at all times, because adaptivity must be preserved to allow the system to follow any characteristic surface variations in time. Also, the main error source in this system is the use of interpolation, as mentioned above.

6. Practical Considerations

In the mechanization of such a scheme in practice, a number of additional issues may need to be considered.

One point to consider is the acquisiton of the measured variables in "steady state" operation. In the example system, for instance, particularly for the thrust and fuel flow rate measurements, a number of consecutive samples will need to be checked to see if their variations are below a certain limit before accepting the measurements as "static", to prevent the engine dynamic transients from corrupting the static characteristics data.

Another point of practical concern is the nonuniform training that will exist during actual jet engine use. This will result in the more frequently encountered regions of the characteristic surface or hypersurface to be trained faster and better, though a buffer memory can be used to store and then repeatedly apply the data from

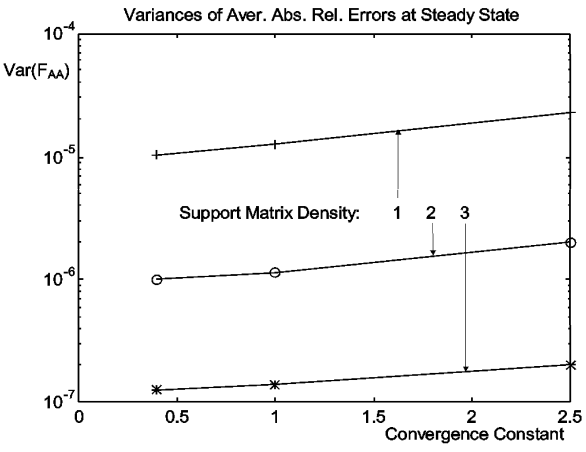

Figure 9. Variance of F_{AA} as a function of D_S and μ.

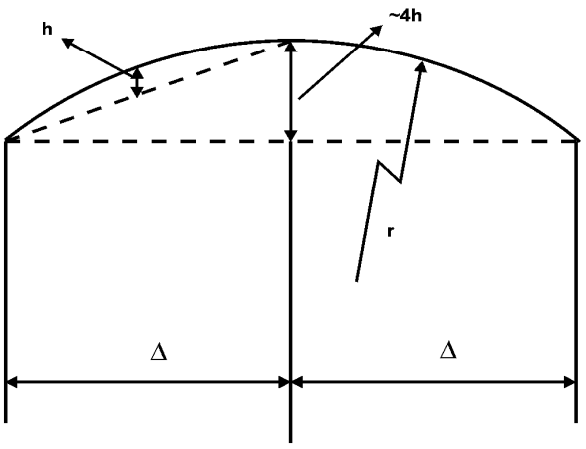

Figure 10. Geometry used in the interpretation of the F_{AA} dependence on cell size.

the less frequently used regions. High sampling rates could obviate this measure, however.

It is possible to use a nonuniformly spaced grid in the input space for defining the support matrix. By making the supports denser where the curvatures of the generated surface are greater, a smaller number of supports can be used for a certain maximum error level. Though this does not seem justifiable for the TSFC surface considered, it may prove useful for some other functions where the curvature may vary over a broad range.

For the example system in particular, the thrust is assumed to be measured directly, yet it is not common for aircraft to have in-flight measurement facilities for thrust, which is usually inferred from other measurements. However, it is a technically feasible measurement (e.g. by strain gages or load cells) and it can be added for such a purpose.

7. Conclusions

The learning algorithm is presented and studied in general, and the automatic derivation and monitoring of a particular characteristic surface for a jet engine is investigated as a usage example. The results of the simulations for the example system give additional clues about the behavior of this learning algorithm.

In particular, the stability, the convergence speed, and the accuracy issues for the algorithm are addressed. A theorem on stability is given. The effects of the support matrix size and the convergence constant on the speed of convergence are considered, and the effects of the cell size and the convergence constant on the accuracy are mentioned. The effect of the convergence constant on the accuracy is related to the noise rejection properties of the algorithm, which is shown to improve as the convergence constant is decreased.

The algorithm has mainly two parameters to decide on: the cell size (or the support matrix density), and the convergence constant. The cell size has greater influence on the steady state accuracies, determining the minimum errors that can be achieved, but an increase in the matrix size will slow down the convergence and require more memory space. With modern digital equipment neither is likely to be a restricting factor since memory sizes of interest are quite small and the sampling rates can be high enough to complete a large number of steps within a short time.

Although the accuracy requirements will vary according to usage, the intermediate support matrix density used in the example system seems to be a suitable choice for this purpose. Choice of the convergence factor will depend on the required learning speed, but in practice, measurement noise is more likely to dictate its value, placing an upper bound for it. On the other hand, too small a convergence constant would cause difficulty in following the variations of the surface in time, especially in case of a rapidly developing anomaly, limiting its value as a post-failure diagnostic aid. Judging in general terms, all convergence constant values tried gave reasonable performance with the example system, with values beyond 3.5 risking instability and values beyond 2.5 having a tendency to create excessive oscillations especially in the transient phase of the learning process where the error factors F are large. The results given here can help as guidelines for a design with a specific purpose.

References

[1] K. J. Åström and B. Wittenmark, Computer Controlled Systems (Prentice - Hall, 1984).

[2] K. J. Åström and B. Wittenmark, Adaptive Control (Addison-Wesley, 1989).

[3] N. A. Hızal, Proc. MAMKON '97, Istanbul Technical University, Faculty of Mechanical Engineering, 496 (1997).

[4] N. A. Hızal, TÜBİTAK Journal of Engineering and Environmental Sciences **22**, 109 (1998).

[5] N. A. Hızal, TÜBİTAK Journal of Engineering and Environmental Sciences **23**, 209 (1999).

[6] P. G. Hill and C. R. Peterson, Mechanics and Thermodynamics of Propulsion (Addison-Wesley, 1967).

[7] J. L. Kerrebrock, Aircraft Engines and Gas Turbines (2nd Edition, The MIT Press, 1992).

[8] J. R. Leigh, Applied Digital Control (Prentice-Hall, 1985).

[9] J. Leimgruber, R. Hribar, and R. Bieri, Sulzer Technical Review **66**, 28 (1984).

[10] J. Leimgruber, P. Lipsky, and A. Reichlin, Sulzer Technical Review **70**, 19 (1988).

[11] J. D. Mattingly, W. H. Heiser, and D. H. Daley, Aircraft Engine Design (American Institute of Aeronautics and Astronautics, 1987).

[12] I. E. Treager, Aircraft Gas Turbine Engine Technology (3rd Edition, McGraw Hill, 1996).