Hybrid Interior Point Methods for Identification of Multiple Local Models of Intelligent Control Systems

Mike Lemmon, Peter T. Szymanski,

Christopher J. Bett, and Panos Antsaklis*

Dept. of Electrical Engineering

University of Notre Dame

Notre Dame, IN 46556

September 20, 1995

Abstract

Intelligent control often uses multiple models of the system to be controlled. These models represent local approximations of the plant which can be used to design switched or supervisory hybrid control systems. This paper describes a recently developed numerical procedure for identifying an optimal set of local system models. The procedure has been shown to have a computational complexity that scales in a polynomial manner with the problem's size. Simulation results illustrating the application of this method to the identification of local models for a fossil fuel electric power plant's input-output behaviour are presented.

*The partial financial support of the National Science Foundation (MSS92-16559) and the Electric Power Research Institute (RP8030-06) are gratefully acknowledged

1 Introduction

Systems engineers routinely face the task of controlling highly complex systems. Examples of such systems include spacecraft, chemical process control, communication networks, and power generation and distribution systems. Such systems are often characterized by a high degree of structural complexity and modeling uncertainty. To deal with these issues, an engineer will often resort to a collection of ad hoc methods whose worth has been proven through empirical experience. The term "intelligent" control has emerged as an umbrella term encompassing many of these methodologies.

One theme uniting many of these "intelligent" control methods is their ability to tackle highly complex systems. Actual engineering systems are complex and must usually be operated in a variety of operational modes to achieve mission objectives. Because these operational modes can possess distinct structural properties, no single mathematical model will be valid over the entire operating range. To adequately control such systems, therefore, it is essential that multiple local models of the system be determined. These local models predict system behaviour in the neighborhood of equilibrium or set points. They can then be used as the basis for switched controller methodologies such as is often found in *supervisory hybrid control systems* (see companion chapter in this book). An important facet of intelligent control is concerned with the determination or extraction of such multiple models. This chapter discusses one particular algorithmic approach to the identification of "optimal" multiple models of dynamical systems.

The problem addressed in this chapter concerns the multiple model parameter identification problem. This problem seeks to identify the parameters of a collection of system models from a large data set that describes the underlying process. This parameter estimation problem can be posed as a non-convex constrained optimization that can be interpreted as a Maximum Likelihood estimation problem. In many cases, the resulting problem can be decomposed into linear and quadratic subproblems with respect to disjoint parameter sets. In this chapter, we use an *alternating minimization* (AM) to solve the non-convex problem. The specific AM procedure discussed below combines interior point (IP) techniques [1,2] and conventional Newton-Raphson methods [3] to solve the linear and quadratic subproblems, respectively. The main contribution of this work is the development of this hybrid IP algorithm and the analytical results concerning its asymptotic convergence and highly attractive computational cost. This algorithm is currently being used to identify multiple local IIR models for power generating systems by post-processing the system's inputs and outputs.

The remainder of this chapter is organized as follows. We first introduce the problem to be solved and pose it as a Maximum Likelihood (ML) estimation problem. The hybrid IP algorithm is then introduced and a number of results are summarized concerning the method's asymptotic behaviour and computational complexity. Empirical results illustrating the algorithm's properties are also discussed. One of the empirical studies involves the identification of multiple local models associated with the control of electric power generating plants. This later example was performed as part of the NSF/EPRI intelligent control initiative.

2 Problem Formulation

In recent years, there have emerged a number of algorithms attempting to identify multiply models of an unknown system. Examples of such procedures consist of neural networks [4–8], fuzzy logic methods [9], genetic algorithms [10], and a host of other popular methods [11–14]. The fundamental problem treated by all of these methods is, essentially, a "non-parametric" function approximation problem. This problem attempts to find an approximation, $\hat{f}$, of an unknown functional, f, by using samples of f's inputs, z, and outputs, y. These input-output samples are called the training set and the objective is to use the training set to find an approximator which is "optimal" in an appropriately defined sense. The so-called "non-parametric" approximation uses approximators of the following form

$$\hat{f}(z) = \sum_{i=1}^{N} \alpha_i k(z|\Theta_i) \tag{1}$$

where $k(z|\Theta_i)$ is a parameterized family of self-similar functions sometimes called kernel or basis functions. Each kernel function can be thought of as an approximator for f which is valid over a local region in the set of inputs. In intelligent control, the unknown mappings, f, are functions of the unknown plant. They may for example be control mappings or flow operators for the original plant. In these applications, the kernel functions constitute local approximations of the plant or control law. Algorithms which efficiently solve these training problems therefore provide an important tool in the design of intelligent control systems.

The hybrid IP algorithm developed in this chapter provides a computationally efficient method for finding "optimal" kernel estimators. In particular, this algorithm determines both the weighting terms, α_i ($i = 1,\ldots,N$), and the kernel parameters, Θ_i ($i = 1,\ldots,N$), with respect to an assumed distortion criterion, $d(f,\hat{f})$, measuring the error between f and its approximation. Since the identification problem involves minimizing this distortion, we can view our problem as an approximation problem. In particular, it has proven very useful to view this as a Maximum Likelihood estimation problem. From this viewpoint, the approximator can be thought of as a "stochastic" generator of outputs whose underlying joint density matches that of the input-output samples. This viewpoint leads to a specific class of bi-convex optimization problems which are particularly well-suited to our hybrid IP algorithm. The remainder of this section discusses this ML viewpoint of the problem in greater detail.

2.1 Stochastic Generator

We'll view the approximator as a stochastic generator. This "stochastic" generator produces outputs, $\hat{y} \in \Re$, in response to inputs, $z \in \Re^R$, where $\hat{y} = \hat{f}(z)$ is an approximation of a bounded, continuous nonlinear function, $f : \Re^R \rightarrow \Re$. The approximator consists of M component generators which generate samples according to a set of underlying probability density functions. The m^{th} probability density function (pdf) is $P(y|m,z,\Theta)$ where Θ is a parameter vector. The generator uses a probability mass function (pmf), $P(m|z,\Theta)$, as a stochastic switch to choose the m^{th} component generator to produce the approximator's output. This approximator is characterized by the parameters, $\Theta = \{\theta_1,\theta_2,\ldots,\theta_M\}$ where $\theta'_m = (\phi'_m,\omega'_m)$. The parameters, (ϕ_m,ω_m), are described below.

Figure 1 illustrates the fundamental architecture being used here. This architecture is also used by a variety of ad hoc neural network [4–7,15] and fuzzy logic [9] modeling paradigms.

Let's consider the output of the m^{th} component generator. This output, $y_m = \hat{f}_m(z)$, is assumed to be produced with respect to a normal density,

$$\begin{aligned} P(y|m,z,\Theta) &= K_e \exp\left(-\frac{1}{\sigma_e^2}\left(f(z)-\hat{f}_m(z)\right)^2\right) && (2)\\ &= K_e \exp\left(-\frac{1}{\sigma_e^2}(y-\phi'_m z)^2\right) && (3)\end{aligned}$$

In the preceding definition, it can be seen that $\phi_m \in \Re^R$ is the parameter vector for the

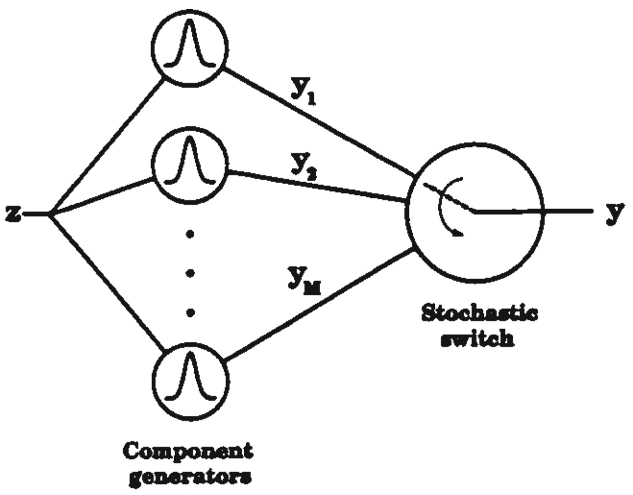

Figure 1: Stochastic generator

component generator's pdf.

The stochastic switch uses the pmf, $P(m|z,\Theta)$, to mix the outputs of the M component generators. The switch constructs the outputs, $\hat{y}$, using a "hard" or strict decision. This means that $\hat{y}$ equals the output of the component generator with the greatest likelihood, $P(m|z,\Theta)$. Using Bayes Theorem and the Theorem of Total Probabilities, this pmf can be expressed as

$$P(m|z,\Theta) = P(z|m,\Theta)\frac{P(m|\Theta)}{P(z|\Theta)} \tag{4}$$

A functional form for $P(z|m,\Theta)$ can be chosen that is consistent with the pdf of the M component generators. In particular, we let

$$P(z|m,\Theta) = K_a \exp\left(-\frac{1}{\sigma_a^2}\|z-\omega_m\|^2\right) \tag{5}$$

From this equation, we see that $\omega_m \in \Re^R$ parameterizes the stochastic switch. ω_m is the mean of the pdf and represents a set point around which the local approximations, $\hat{f}_m(z)$, are linearized.

2.2 Mixture Density Parameter Estimation

The specific problem of interest is discussed in this subsection. Consider a collection of inputs and outputs for a function, f, which we will call the training set. This collection is denoted as $\mathcal{T} = \{(y_j,z_j) : y_j = f(z_j)\}$. For a given stochastic generator with parameterization, Θ, the objective will be to find the parameters, Θ^*, that maximize the expected likelihood of the stochastic generator in producing the training samples.

This "likelihood" measure can be formulated as follows. Let $P(y,m|z,\Theta)$ be the likelihood of generating an output, y, using the m^{th} component generator given an input, z and a parameterization, Θ. The *log likelihood* function, $l(y,m|z,\Theta) = \log P(y,m|z,\Theta)$, can be expressed as

$$\begin{aligned} l(y,m|z,\Theta) &= \log P(y|m,z,\Theta) + \log P(m|z,\Theta) & (6) \\ &= \log P(y|m,z,\Theta) + \log P(z|m,\Theta) + \log \frac{P(m|\Theta)}{P(z|\Theta)} & (7) \end{aligned}$$

Using the assumed functional forms of the component generator's pdf and the stochastic switch's pmf, this likelihood becomes

$$l(y,m|z,\Theta) = -\frac{1}{\sigma_e^2}(y-\phi_m^T z)^2 - \frac{1}{\sigma_a^2}\|\omega_m - z\|^2 + \log K_e K_a \frac{q(m)}{P(z)} \quad (8)$$

where $q(m)$ is the likelihood of using the m^{th} component generator and $P(z)$ is the probability of input z being observed. Recognizing that both of the likelihoods are independent of the parameterization, we can neglect the last term above and obtain the simpler form

$$l(y,m|z,\Theta) = -\frac{1}{\sigma_e^2}(y-\phi_m^T z)^2 - \frac{1}{\sigma_a^2}\|\omega_m - z\|^2 \quad (9)$$

This quantity is, of course, a sum of squared error measures on the inputs and outputs of the function.

Note that $l(y,m,|z,\Theta)$ is a random variable. We examine the problem of maximizing the expected value (over z and m) of this likelihood function with respect to Θ. Note, however, that since we will be working with a finite training set, the actual optimization will be with respect to an ensemble average of the expected likelihood. This ensemble average of the likelihood function is expressed as

$$E_T\left[l(y,m|z,\Theta)\right] = \sum_{j=1}^{N}\sum_{m=1}^{M} P(m,z_j|\Theta) l(y,m|z,\Theta) \quad (10)$$

where $P(m,z|\Theta)$ is the joint likelihood of using the m^{th} component generator and observing input z. The objective, therefore, is to minimize the above quantity with respect to Θ.

One approach for solving this problem is to use an Expectation-Maximization [16,17] (EM) procedure. The EM method solves this problem by iteratively computing an expectation that depends upon the current parameterization of the problem, Θ', and then by maximizing the expectation with respect to Θ to determine a new Θ'.

1. **Expectation**

$$E_T\left[l(y,m|z,\Theta)\right] = \sum_{j=1}^{N}\sum_{m=1}^{M} P(z_j|\Theta')P(m|z_j,\Theta')l(y,m|z,\Theta)$$

2. **Maximization**

$$\Theta' = \arg\max_{\Theta}\sum_{j=1}^{N}\sum_{m=1}^{M} P(z_j|\Theta')P(m|z_j,\Theta')l(y,m|z,\Theta)$$

The procedure produces a sequence of parameter estimates that converges to optimal ML parameter estimates [16,18,19].

By viewing the above problem through the eyes of an EM iteration, we can formulate a slightly different constrained optimization problem which can be addressed using the hybrid IP methods described in this chapter. The alternate form is obtained by recognizing that $P(m,z_j|\Theta') = P(z_j|\Theta')P(m|z_j,\Theta')$ in the EM procedure is independent of Θ. We can therefore remove the dependence upon Θ and rewrite the marginal and conditional densities as $p(z_j)$ and $Q(m|j)$, respectively. The conditional probabilities, $Q(m|j)$, change with every iteration due to changes in Θ'. If the probabilities functional dependence upon Θ' is removed, then the change in the $Q(m|j)$'s can be controlled by maximizing the expectation with respect to Q after the maximization with respect to Θ. Using this viewpoint, we are led to a maximization problem of the following form

$$\begin{array}{rl} \text{minimize:} & \sum_{j=1}^{N}\sum_{m=1}^{M} p(j)Q(m|j)\left[\frac{1}{\sigma_e^2}(y_j-\phi_m^T z_j)^2 + \frac{1}{\sigma_a^2}\|z_j-\omega_m\|^2\right] \\ \text{with respect to:} & Q,\Theta \\ \text{subject to:} & \sum_{m=1}^{M} Q(m|j) = 1, \ \text{for } j=1,\ldots,N \\ & Q(m|j) \geq 0 \end{array} \tag{11}$$

The constraints force Q to be a valid set of pmf's. This is necessary because the functional form of the densities and the dependence upon Θ are not explicitly shown. The problem can be solved iteratively by alternately solving first with respect to Q and then with respect to Θ.

The preceding problem formulation is a constrained optimization problem possessing the bi-convex nature demanded by our algorithm. This problem is found very often in a number of disciplines [4,12,20,21]. In the context of "intelligent" control, this problem formulation arises from the Saridis formulation of hierarchical intelligent control [12], in which entropy measures are used to construct a hierarchical control system in which greater "intelligence" is concentrated in the higher levels of the hierarchy. In this chapter, we interpret the problem in the context of traditional function approximation theory. The function to be approximated is a complex, possibly nonlinear, plant for which we are seeking a fixed number of locally valid models. The objective is then to select an optimal set of such models.

3 Alternating Minimization

This section discusses the hybrid IP algorithm. The preceding problem falls into that class of problems that can be decomposed into linear and quadratic subproblems with respect to disjoint parameter sets. The hybrid IP algorithm for solving this class of constrained optimization problems arises from the combination of a small step path following algorithm [1] and Newton-Raphson techniques [3] to solve the linear and quadratic subproblems, respectively. A basic description of the hybrid IP algorithm appears in this section along with a discussion of the issues that must be addressed to guarantee its convergence. Proofs of the parameter choices, convergence, and complexity issues will be found in [22, 23] and are only summarized below.

The original problem can be rewritten in a form that emphasizes both its objective functional's dependence upon two disjoint sets of parameters and highlights the connection to linear programming. Define the vectors

$$x = (Q(1|1), \ldots, Q(1|N), Q(2|1), \ldots, Q(m|j), \ldots)' \tag{12}$$

and

$$c(\Theta) = (p(1)d_1(1), \ldots, p(N)d_1(N), p(2), d_2(2), \ldots, p(j)d_m(j), \ldots)' \tag{13}$$

with $d_m(j)$ being our distortion measure. Using this notation, the problem in the preceding section is written as a perturbed linear programming problem

$$\begin{array}{rl} \text{minimize:} & c(\Theta)'x \\ \text{with respect to:} & x, \Theta \\ \text{subject to:} & Ax = b \\ & x \geq 0 \end{array} \tag{14}$$

where $A = [I_{N\times N} \ldots I_{N\times N}]$, $b = (1, 1 \ldots, 1)'$, and $x \geq 0$ means that each component of the vector is non-negative. The objective functional is the inner product of two vectors which depend upon unique and disjoint sets of parameters. The linear nature of the optimization problem is obvious from the form of the problem's objective functional. The problem is an LP problem with respect to x. Alternately, when the objective functional is considered to be solely a function of Θ, then we have a quadratic optimization with respect to Θ.

The dual nature of the problem suggests its solution via an alternating minimization. Minimizing the problem with respect to x and Θ simultaneously will result in locally optimal solutions due to the non-convexity of the problem; and optimization using standard techniques may take some time to converge. The problem, however, can be decomposed into two convex subproblems: the linear programming (LP) subproblem with respect to x and the quadratic subproblem with respect to Θ. The two convex subproblems can be solved efficiently using interior point (IP) techniques to solve the LP subproblem and Newton-Raphson (NR) techniques to solve the quadratic subproblem. An appropriate combination of the two methods results in an iterative algorithm that produces locally optimal solutions in an efficient manner. That is precisely what is done in the hybrid IP algorithm presented below. The following paragraphs describe the linear and quadratic problem solvers, and then describe the hybrid IP algorithm.

A small step path following algorithm solves the optimization with respect to x. The path following algorithm solves the problem with respect to x by solving the following sequence of optimization problems for $k = 1, 2, \ldots$ and $\alpha(k) \geq 0$

$$\begin{aligned} \text{minimize:} \quad & \alpha^{(k)} c(\Theta)' x - \textstyle\sum_i \log x_i^{(k)} \\ \text{with respect to:} \quad & x \\ \text{subject to:} \quad & Ax = b \end{aligned} \tag{15}$$

$\alpha^{(k)} \geq 0$ is a finite length monotone increasing sequence of real numbers and $\sum_i \log x_i^{(k)}$ is a set of logarithmic barriers which tend to keep intermediate solutions away from the boundary of the set of feasible solutions. $x^*(\alpha^{(k)})$ denotes the optimal solution for the kth problem and is referred to as the kth central point. The locus of all central points $x^*(\alpha^{(k)})$ is called the central path. As $\alpha^{(i)}$ increases, the effects of the barriers decrease and the sequence of central points approach the optimal LP solution which lies on the boundary of the feasible set.

The path following algorithm approximately solves each problem in the sequence using a

Scaling Steepest Descent (SSD) update [1]

$$x^{(k+1)} = x^{(k)} - X^{(k)} P_{AX} X^{(k)} (\alpha^{(k+1)} c - (x^{(k)})^{-1}) \tag{16}$$

where $X = \mathrm{diag}(x_1, x_2, \ldots, x_n)$ is a scaling transformation, $P_A = I - A^T(AA^T)^{-1}A$ is an orthogonal projection matrix and $x^{-1} = (x_1^{-1}, \ldots, x_n^{-1})'$. Assuming that $x^{(k)}$ is near to the kth central point and by choosing $\alpha^{(k)}$ as $\alpha^{(k+1)} = \alpha^{(k)}(1 + \nu/\sqrt{n})$ where $\nu \in (0, 0.1]$, $x^{(k+1)}$ can be found near to the $(k+1)$st central point in a single SSD update. The sequence of solutions, $\{x^{(k)}\}$, follows the central path the to the optimal LP solution as $\alpha^{(k)}$ increases. The algorithm runs until the duality gap, $\Delta^{(k)}$, for the feasible LP problem falls below a prespecified tolerance. The basic small step path following algorithm is shown below.

Algorithm 1 (Basic path following algorithm)

```
Initialize
    $k = 1$.
    Choose $\alpha^{(k)} > 0$ and initial feasible $\mathbf{x}^{(k)}$.
    Centralize $\mathbf{x}^{(k)}$ for $\alpha^{(k)}$.
repeat
    $\alpha^{(k+1)} = \alpha^{(k)}(1 + \nu/\sqrt{n})$, where $\nu \in (0, 0.1]$.
    $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - X^{(k)} P_{AX^{(k)}} X^{(k)} (\alpha^{(k+1)} \mathbf{c} - (\mathbf{x}^{(k)})^{-1})$
    $k = k + 1$
until($\Delta^{(k)} < \epsilon$)
```

A Newton-Raphson (NR) approach solves the quadratic optimization with respect to Θ. The basic form of an NR update is

$$\Theta^{(k+1)} = \Theta^{(k)} - H(\Theta^{(k)}) G(\Theta^{(k)}) \tag{17}$$

where $H(\Theta)$ is the Hessian of the objective functional with respect to Θ and $G(\Theta)$ is the gradient with respect to Θ. Since the optimization is quadratic, the following closed form solution results from the NR update for the optimization with respect to Θ. The optimum, Θ^*, is the vector $(\theta_1^*, \ldots, \theta_M^*)$ where $\theta_m^* = (\phi_m^*, \omega_m^*)$ and

$$\phi_m^* = \left[\sum_{j=1}^{N} p(j) Q(m|j) z_j z_j^T \right]^{-1} \sum_{j=1}^{N} p(j) Q(m|j) z_j y_j \tag{18}$$

and

$$\omega_m^* = \frac{\sum_{j=1}^{N} p(j) Q(m|j) z_j}{\sum_{j=1}^{N} p(j) Q(m|j)} \tag{19}$$

The aforementioned two methods combine to form the hybrid IP algorithm for performing alternating minimizations. The NR minimizer which is called the Θ-update is embedded in the path following algorithm following the SSD update, which is termed the Q-update. The algorithm computes a sequence of solutions, $\{x^{(k)}, \theta^{(k)}\}$, that converges to an optimal parameter set. An iteration of the algorithm incrementally adjusts first the linear parameters and then the quadratic parameters. The actual hybrid IP algorithm appears below after a brief discussion of the issues to be addressed in its application.

A key concern in the basic path following algorithm is the proximity of intermediate solutions in the sequence $\{x^{(k)}\}$ to the central path. The nearness of successive solutions to the central path is paramount to ensure algorithm convergence as well as its attractive computational complexity. For the basic IP algorithm, such proximity guarantees that the algorithm will find the LP solution after $O(\sqrt{n}L)$ iterations with a total cost of $O(n^{3.5}L)$ where n is the LP dimension and L is the size of the LP problem [1]. This "proximity" condition must also be maintained by the hybrid IP algorithm if we are to guarantee convergence and low computational complexity. The proximity condition is stated below

$$\begin{aligned} \|P_{AX}X^{(k)}(\alpha^{(k)}c(\Theta) - (x^{(k)})^{-1})\| &= \delta(x^{(k)}, \alpha^{(k)}, \Theta^{(k)}) && (20) \\ &\leq 0.5 && (21) \end{aligned}$$

Ensuring the proximity of successive solutions to the central path is accomplished by constraining the quadratic optimization in the hybrid IP algorithm. Minimizing $c(\Theta)'x$ with respect to Θ within the path following algorithm changes the cost vector, $c(\Theta)$. It is possible that such changes in the cost vector will rotate the cost about the current solution in such a way that $x^{(k)}$ no longer satisfies the nearness condition. In order to ensure that the condition is met, it is necessary to be less aggressive in the Θ-update. In particular, we constrain the NR iteration by updating Θ as the convex combination of the current and minimizing parameter vectors

$$\Theta^{(k+1)} = (1-\gamma^{(k)})\Theta^{(k)} + \gamma^{(k)}\Theta^{(k+1),*} \qquad (22)$$

where $\gamma^{(k)} \in (0, \gamma_{\text{bound}}]$. A major accomplishment of this work was the derivation of γ_{bound} that guarantees this proximity condition.

A second concern arises from the use of the constrained Θ-update just described. The Θ-update requires that $x^{(k)}$ actually be closer to x^* than required by the proximity condition.

This extra proximity is necessary as the Θ-update rotates the cost vector and can increase the norm. In the following algorithm, this extra proximity condition is met by performing two SSD updates rather than the single update used in the basic IP method.

These two concerns are addressed in the algorithm shown below. In the following section, specific parameter choices are given and existing results on the method's computational complexity and asymptotic convergence are summarized.

Algorithm 2 (Alternating minimization algorithm)

```
Initialize
    $k = 0$.
    Choose $\alpha^{(k)} > 0$, $\Theta^{(k)}$, and initial feasible $\mathbf{x}^{(k)}$.
    Centralize $\mathbf{x}^{(k)}$ for $\alpha^{(k)}$.
repeat
    $\alpha^{(k+1)} = \alpha^{(k)}(1 + \nu/\sqrt{n})$, where $\nu \in (0, 0.1]$.
    $Q$-update:
        $\mathbf{x}_0 = \mathbf{x}^{(k)}$
        for $i = 0, 1$
            $\mathbf{x}_{i+1} = \mathbf{x}_i - X_i P_{AX_i} X_i (\alpha^{(k+1)} \mathbf{c}(\Theta^{(k)}) - \mathbf{x}_i^{-1})$
        $\mathbf{x}^{(k+1)} = \mathbf{x}_2$
    $\Theta$-update:  For $m = 1, \ldots, M$, where $\gamma^{(k)} \in (0, \gamma_{bound}]$
            $\theta_m^{(k+1)} = (1 - \gamma^{(k)})\theta_m^{(k)} + \gamma^{(k)}\theta_m^{(k+1),*}$
    $k = k + 1$
until($\Delta^{(k)} < \epsilon$)
```

4 Summary of Hybrid IP Results

Successful operation of the hybrid IP algorithm requires the appropriate choice for a number of parameters. The parameters, $\gamma^{(k)}$ and ν, and the number of SSD steps to take in the Q-update are the free parameters characterizing this algorithm. It must then be shown that there exist parameters which yield a convergent algorithm preserving the low computational complexity found in basic IP algorithms. This section summarizes existing results concerning these methods. The results have been proven in the cited papers.

The first proposition of interest states that 2 SSD steps are required for to meet the extra stringent proximity condition. This proposition also provides bounds on the Q-update's step size in terms of a bound on ν, and on the extent of the Θ-update in terms of a bound on

$\gamma^{(k)}$. Provided these conditions are met, then the hybrid IP algorithm always satisfies the proximity condition introduced above.

Proposition 1 *Let* $\delta(x^{(k)}, \alpha^{(k)}, \Theta^{(k)}) \leq 0.5$ *and let* $\alpha^{(k+1)} = \alpha^{(k)}(1 + \nu/\sqrt{n})$ *where* $\nu \in (0, 0.1]$. *Let* $\Theta^{(k+1)} = (1 - \gamma^{(k)})\Theta^{(k)} + \gamma^{(k)}\Theta^{(k+1)}\Theta^{(k+1),*}$ *where*

$$\gamma^{(k)} \leq \frac{0.12995}{n\|c^{(k+1),*} - c^{(k)}\|} \tag{23}$$

Then one iteration of the hybrid IP algorithm with two SSD steps produces a $(x^{(k+1)}, \Theta^{(k+1)})$ *such that* $\delta(x^{(k+1)}, \alpha^{(k+1)}, \Theta^{(k+1)}) \leq 0.5$.

With the preceding proposition, it can now be shown that the sequence of solutions generated by the hybrid IP algorithm will asymptotically converge to a locally optimal solution of our ML problem. This result rests on the fact that the distance to the local optimum of the bi-convex problem can be bounded by the duality gap of the linear programming part of the problem. In particular, the relationship between changes in the Θ and Q-updates are related as follows

$$\|\Theta^{(k+1),*} - \Theta^{(k),*}\| \leq M\mathcal{K}\|\Delta Q\| \tag{24}$$

where $\mathcal{K}$ is a proportionality term dependent on the eigenvalues of the training set's autocorrelation matrix. The formal technical results are proven in the cited references. With the above relationships established the following theorem can then be proven.

Proposition 2 *For the hybrid IP algorithm, there exists* $K \geq 0$ *such that* $\gamma^{(k)} = 1$ *for all* $k \geq K$. *For all* $k \geq K$, *the sequence of solutions converges to a fixed point,* (x^*, Θ^*) *which is a local minimum of the ML problem stated above.*

The preceding convergence results are asymptotic in nature. To be useful, however, we need to study the computational complexity of this procedure. Remember, the reason for introducing a hybrid IP algorithm was that with appropriate scaling it might inherit the attractive computational complexity of the basic IP procedure. The following proposition states that with appropriate scaling, the hybrid IP algorithm does indeed inherit the attractive complexity properties of its basic IP counterpart.

Proposition 3 *Let* $w^{(k)}$ *be the kth approximate solution generated by the hybrid IP algorithm. Assume that* $\Delta^{(0)} < 1/\epsilon$. *Assume that the cost vector is scaled such that* $\gamma^{(k)} = 1$ *for all*

$k \geq K$ and that $2(\delta(1+\nu)+\nu)(1+MK)\Delta^{(K)} > \epsilon$. Given these assumptions,the hybrid IP algorithm will converge to an ϵ-neighborhood of a locally optimal solution in $O(\sqrt{n}\log_2(\sqrt{n}/\epsilon))$ iterations.

The hybrid IP algorithm's convergence rate is slightly worse than its basic path following counterpart. The basic IP algorithm converges in $O(\sqrt{n}L)$ iterations where L is the size of the LP problem. L may be interpreted as $\log_2(1/\epsilon)$, where ϵ specifies a neighborhood around the optimal LP cost. If it is taken as such, the rates differ by $\log_2\sqrt{n}$ term. The difference is due to the fact that the path following algorithm is converging to an ϵ-neighborhood of the optimal cost solution, $c'x^*$ while the hybrid IP algorithm converges to the ϵ-neighborhood of the optimal parameter solution (x^*, Θ^*). In cases where $\sqrt{n}$ is small compared to $1/\epsilon$ then the hybrid IP algorithm exhibits nearly the same convergence rate as its basic IP counterpart.

The hybrid IP algorithm's overall computational cost is computed by determining the computational complexity of individual steps and then multiplying by the algorithm's convergence rate. The Q-update executes two SSD steps each employing a matrix inversion with worst case complexity of $O(n^3)$ where $n = MN$. The Θ-update requires NR^2 multiplications resulting from the vector outer product and employs a matrix inversion with a computational complexity of $O(R^3)$. Also, the Θ-update is done on M models so this step has a total complexity of $O(MR^3)$. These two complexity estimates yields the following theorem.

Proposition 4 *The computational cost of the hybrid IP algorithm is $O((n^{3.5} + n^{1.5}R^2 + \sqrt{n}MR^3)\log_2(\sqrt{n}/\epsilon)$.*

The preceding results suggest the use of the duality gap as a stopping criterion for the hybrid IP algorithm. The distance to the optimal solution is bounded by a scaled version of the duality gap

$$\|w^{(k)} - w^*\| < 2(\Delta(1+\nu)+\nu)(1+MK)\frac{\delta^{(k)}}{1-\beta} \tag{25}$$

In order to reach the neighborhood of an optimum specified by ϵ, the duality gap, $\Delta^{(k)}$, must be reduced until

$$\Delta^{(k)} < \frac{1-\beta}{2(\delta(1+\nu)+\nu)(1+MK)}\epsilon \tag{26}$$

If the duality gap is reduced below this value, then the parameter sequence will have converged to an ϵ-neighborhood of w^*. Thus the duality gap can be used as a stopping criterion for the hybrid IP algorithm.

5 Examples

The hybrid IP algorithm was developed under the sponsorship of the NSF and EPRI. One objective of this project was to provide a method for the fast and efficient determination of multiple local models of complex dynamical systems. This objective was considered important to intelligent control. Intelligent control seeks methods for the regulation and supervision of complex systems. Due to complexity, such systems can rarely be accurately represented with a single mathematical model. The "intelligent" control of such systems therefore requires several mathematical models of the system. The domain of intelligent control concerns the use and coordination of these various models in regulating the plant. Obviously, one of the first steps in the process is the extraction of an optimal set of local system representations. The hybrid IP algorithm appears to provide a very powerful tool for accomplishing this goal.

In this section, we describe three applications of the hybrid IP procedure. All three applications use the hybrid IP algorithm to identify a fixed set of models which are locally valid. The objective of these experiments was to validate the previously summarized analytical results and then to illustrate their use on an application pertaining to a problem posed by the Electric Power Research Institute. This last part of the section constitutes that part of the work which was funded under sponsorship of the NSF/EPRI intelligent control initiative. Software developed for this project was delivered to a company (ATI, Inc.) associated with an EPRI sponsored project. This project was to provide a software package that could assist plant operators in fine-tuning the performance of a load following plant.

5.1 Comparison to Other Methods

This subsection discusses a comparison of the hybrid IP algorithm with the Expectation-Maximization (EM) procedure [16, 17]. The following results show that the hybrid IP algorithm exhibits a high initial cost, thereby making it unsuitable for small problems. For larger problems, however, the sublinear scaling of the iteration count with respect to problem size makes the hybrid IP algorithm highly attractive. The following results also demonstrate that the complexity and convergence results are very independent of problem structure and initial conditions.

In this problem, we consider the problem of approximating an unknown probability density function using the hybrid IP algorithm. We assume that the training set consists of samples drawn from a multi-modal normal distribution. The objective is to find a suitable density approximation using the basic EM algorithm and using the hybrid IP procedure. The problem is equivalent to a Mixture of Densities Parameter Estimation problem [16,17].

Figures 2 and 3 illustrate the basic nature of the results obtained in the comparison. The results examine the computational properties versus the number of models or densities in the mixture. In the figures, the hybrid IP algorithm's points are denoted by x's and the EM algorithm's points by open circles. The figures plot the empirical convergence rates and costs with a "best fit" characteristic for both methods. The hybrid IP algorithm has a convergence rate and computational cost of $O(\sqrt{M})$ iterations and $O(M^2)$ floating point operations, respectively. The estimated rates for the EM algorithm were $O(M^{2.4})$ iterations and $O(M^{3.4})$ operations.

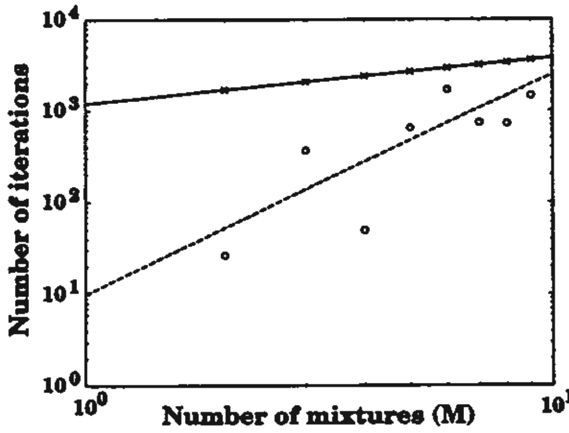

Figure 2: Convergence rate

Extensive simulation experiments which appear in [22] indicated that these trends held consistently. In particular, it was noted that the EM algorithm exhibited a cost which was highly dependent upon the nature of the problem as well as the initial conditions. The exponents noted above varied from problem to problem. Additionally, we saw that the number of iterations for the same problem varied considerable based on the number of models sought and on the initial conditions. In contrast to this, the hybrid IP algorithm exhibited a complexity whose scaling was consistently the same from problem to problem.

M.D. Lemmon, P.T. Szymanski, C.J. Bett, and P.J. Antsaklis, "Hybrid Interior Point Methods for Identification of Multiple Local Models of Intelligent Control Systems," Technical Report of the ISIS (Interdisciplinary Studies of Intelligent Systems) Group, No. ISIS-95-009, Univ of Notre Dame, September 1995. Chapter in I ntelligent C ontro l .

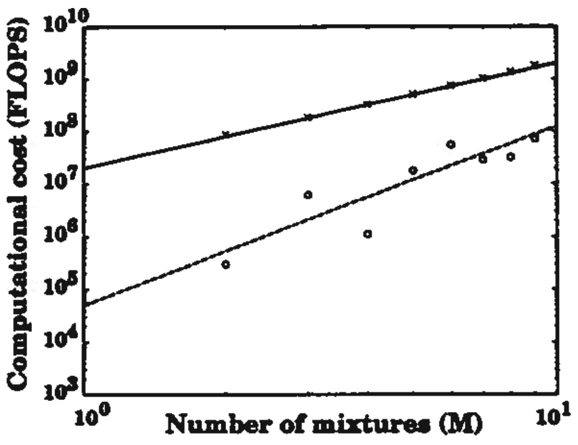

Figure 3: Computational cost

5.2 Local Linear FIR Models of Time Series

This subsection discusses experiments using the hybrid IP algorithm to identify a fixed set of local FIR models generated from a function which is the output of a discrete-time dynamical system. The test dynamical system is

$$\begin{aligned} \mathbf{z}[n+1] &= A[n]\mathbf{z}[n] + \mathbf{b}u[n] \\ y[n] &= \mathbf{c}[n]^T\mathbf{z}[n] \end{aligned} \tag{27}$$

where

$$\begin{aligned} A[n] &= \begin{bmatrix} 2e^{-(\alpha_0+\alpha)T}\cos(2\pi fT) & -e^{-2(\alpha_0+\alpha)T} \\ 1 & 0 \end{bmatrix} \\ \mathbf{b} &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} c_1 e^{-(\alpha_0+\alpha)T}\sin(2\pi fT) \\ 0 \end{bmatrix} \end{aligned} \tag{28}$$

with $\alpha_0 = 1$, $f = 5$, $T = 0.01$ seconds, $c_1 = 0.3249$, and $\alpha = 10|y[n]|$. This corresponds to a structured perturbation of a linear system with impulse response

$$h(nT) = e^{-\alpha nT}\sin(2\pi fnT). \tag{29}$$

The specific signal used in the experiments to identify the parameters, which appears in Figure 4, is the output of the dynamical system in response to

$$u[n] = 10\sin(\pi nT) + 10\sin(3\pi nT) + 10\sin(5\pi nT). \tag{30}$$

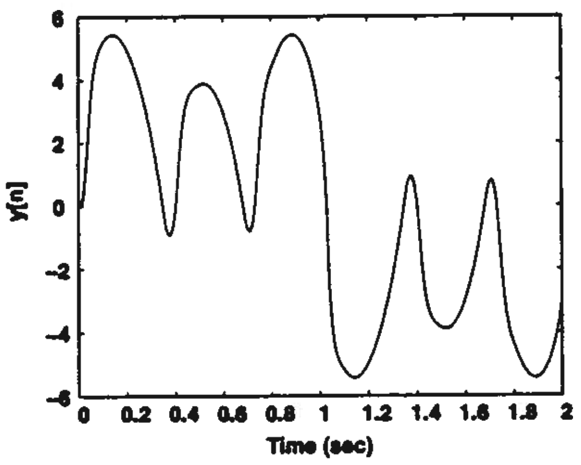

Figure 4: Test signal

The goal of the time series prediction problem is to determine a prediction, $\hat{y}[n]$, of a function, $y[n]$, such that the mean square error is small. This is done using a window of R past outputs, $\mathbf{z}[n] = (y[n-1], y[n-2], \ldots, y[n-R])^T$. A network of linear regressors [22,23] is used to compute the prediction. The m^{th} regressor is an FIR filter parameterized by the vector $\phi_m \in \Re^R$ and produces an output as $y_m[n] = \phi_m^T \mathbf{z}[n]$. The problem assumes a training set, $\mathcal{T} = \{(y_i, \mathbf{z}_i)\}$, of $N = 100$ random samples of the output of (27) over the range, $nT \in [0, 2]$ seconds. The samples are used by the hybrid IP algorithm to identify the regressor parameter vectors.

The results for this example examine parameter identification for the case where a window of past outputs of size $R = 8$ is used. Figure 5 displays the convergence rate of the hybrid IP algorithm for $M = 2$ to 9 models. The empirical results (open circles) agree very well with the predicted theoretical convergence rate of $O(\sqrt{n}\log_2(\sqrt{n}/\epsilon))$ iterations (solid line). Figure 6 depicts the computational cost of the algorithm versus the number of models. In this case, the empirical cost (open circles) agrees more closely with a computational cost of $O(n^2\log_2(\sqrt{n}/\epsilon))$ floating point operations (FLOPS) (solid line) than with the predicted theoretical rate of $O(n^{3.5}\log_2(\sqrt{n}/\epsilon))$ FLOPS (dashed line). This discrepancy is due to the fact that sparse matrix techniques were used to implement the SSD update.

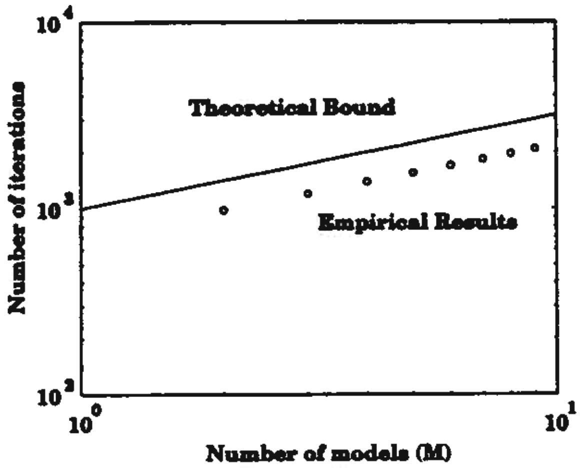

Figure 5: Convergence rate example 1

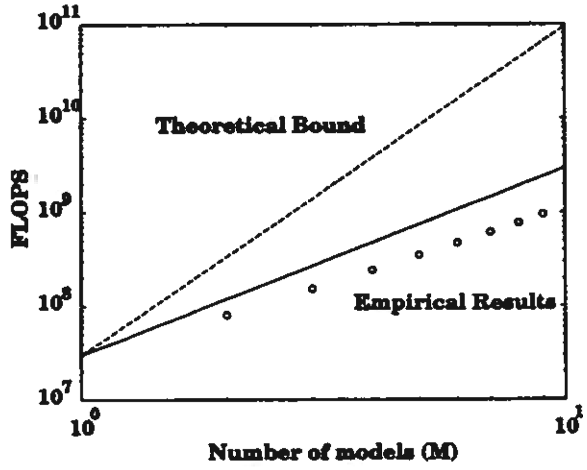

Figure 6: Computational cost example 1

5.3 Power Generating Plant Example

This subsection discusses the hybrid IP algorithm's use in post-processing input-output data from the generating plant to identify multiple local models of parts of the plant. This application was studied as a part of the EPRI/NSF intelligent control initiative. The objective was to develop a collection of software tools implementing the hybrid IP algorithm. These software tools would be used in a commercial product being developed by a private firm (ATI Inc.) to assist plant operators at fossil fuel power plants. The following results were generated using the software delivered to ATI Inc. and are illustrated on data obtained from an actual load following power plant. The results show that the algorithm does exhibit the predicted polynomial properties and that it determines a set of local models which are switched when used to model the plant.

This problem examines the fuel flow model of the plant. The fuel flow dynamics describe the *heat release*, or the inferred fuel flow to a furnace calculated as steam flow added to the rate of change of drum pressure, in response to the fuel controller's output. Figures 7(a) and 7(b), respectively, depict the input (fuel controller output) and the output (heat release). Both quantities appear as a percentage of the maximum allowed fuel controller output and heat release for the plant. The data for input and output consists of 86399 sample points where the input and output were sampled every second over the course of a day.

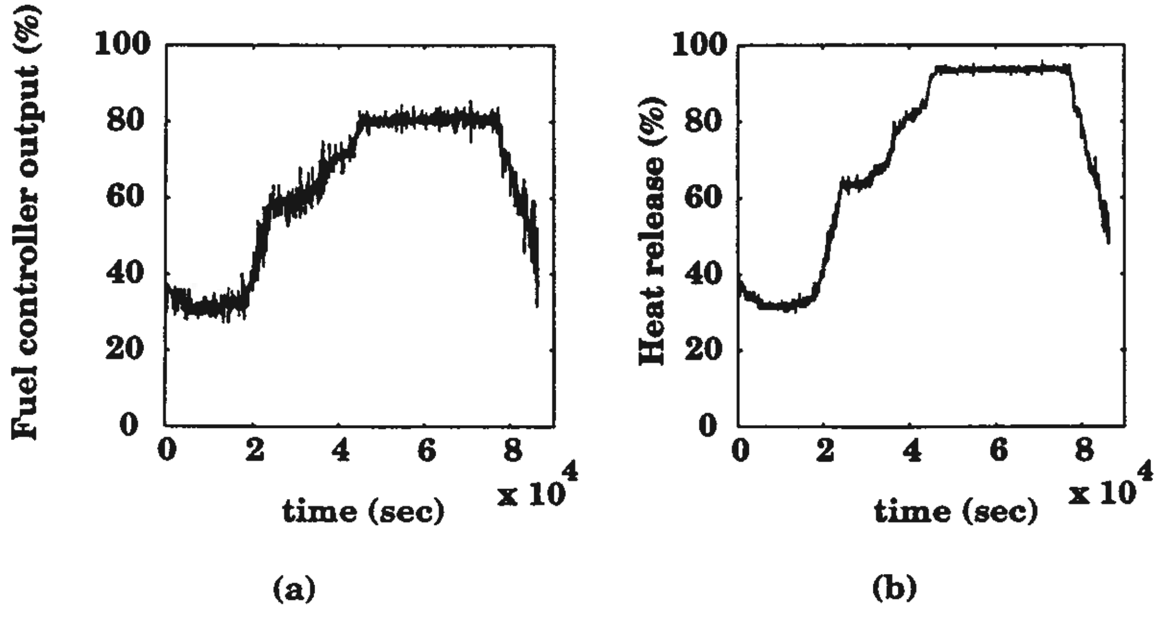

Figure 7: Input-Output Fuel Flow Data

The data depict the trends in the input and output, but have noise associated with them. The level of noise in the input and output is determined by finding the trends in the data and then determining the noise characteristics of the actual data with respect to the trended data. Table 1 tabulates the mean noise and noise variance for the input and output.

Table 1: Input-Output Data Noise Characteristics

Data	Mean noise	Noise variance
Input	4.6066	4.6066
Output	1.0474	1.0473

The fuel flow data experiments used a data set of $N = 500$ points for training. The data set consisted of points, $\mathcal{T} = \{(y_j, z_j) : j = 1, \ldots, 500\}$, where $y_j = y(t_j)$ and $z_j = (y(t_j - 60), y(t_j - 120), \ldots, y(t_j - 540), x(t_j), x(t_j - 60), \ldots, x(t_j - 540))'$ where $y(t)$ is the output at time t and $x(t)$ is the input at time t. The points correspond to pairs where ten inputs and nine past outputs are used to predict the current output. The points that make up the z_j's consist of successive points which were sampled every 60 seconds from the data.

The models identified by the experiments are local IIR filters. The filters are 10^{th} order, having ten taps for the inputs and 9 taps for the past outputs. The filter coefficients are training using the hybrid IP algorithm and the data set described above.

The results of the experiments examine the algorithm's computational complexity and the resulting models. Figures 8 and 9 show the convergence rate and computational cost incurred by the algorithm in training the models. As with the previous examples, the convergence rate agrees with the $O(\sqrt{n}\log_2(\sqrt{n}/\epsilon))$ theoretical rate. The cost is again better than the theoretical rate. In this case, however, the cost is $O(n^{1.6}\log_2(\sqrt{n}/\epsilon))$ operations, which is better than the theoretical rate and the previous examples. This is due to the sparse implementation and the fact that n in these experiments is much larger than R.

The resulting models are validated by examining the mean square error of the models when they are used to approximate the input-output relation. A set of $N = 500$ points serves as the validation set. The mean square error over the validation set as a function of the number of models appears in Figure 10. It shows that the resulting models have mean square errors that are on the order of the variance of the noise. The general trend in the plot shows that the mean square error increases with the number of models. This is

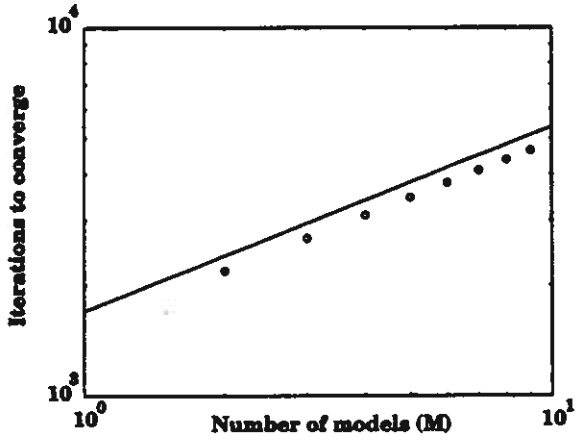

Figure 8: Convergence Rate — Fuel Flow Data

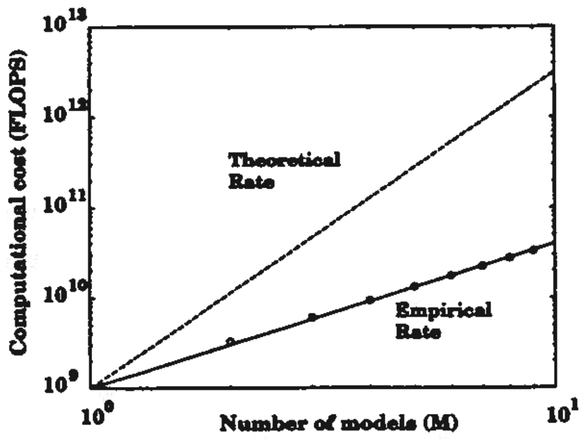

Figure 9: Computational Cost — Fuel Flow Data

interpreted as meaning that the models are complex enough that a single 10th order model would accurately represent the system. The increase in mean square error, then, is due to the variance of the noise as is expected when training the models. Other experiments are currently under way that examine lower order models.

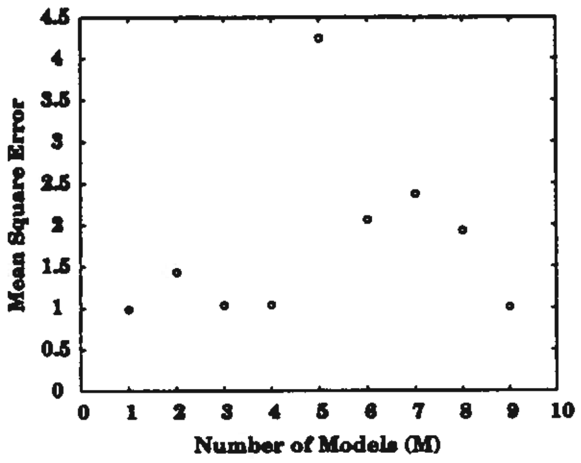

Figure 10: Mean Square Error — Fuel Flow Data

A demonstration of the switching of the models appears in Figure 11. The plot depicts the output for an ensemble of $M = 3$ models. The ensemble's output is the output of individual model that has the greatest likelihood of being the best approximator for the current input, as discussed above. The points which correspond to the three models are plotted as x's, open circles, or dots. The plot demonstrates that the models operate over local regions of the input and output space, which was the goal in using the hybrid IP algorithm.

6 Summary

This chapter has introduced a new interior point (IP) algorithm that can be used to solve a wide class of problems. The particular problem considered in this chapter revolved about the identification of multiple local models of a power generating plant. The algorithm developed under this project is, to the best of our knowledge, the first such algorithm developed for solving Maximum Likelihood problems with a generalized Expectation-Maximization procedure whose convergence and complexity properties have been formally characterized. The development of such an algorithm for the special class of problems considered here immediately suggests that such methods may be applicable to solving a variety of related problems:

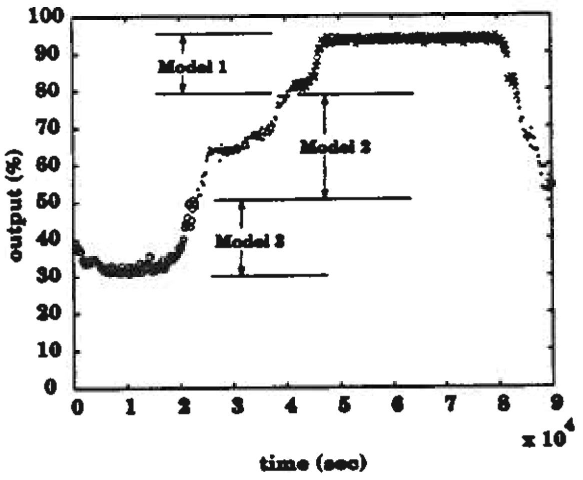

Figure 11: Model Switching — Fuel Flow Data

automatic speech recognition [20, 24], hierarchical intelligent control [12], hidden Markov model identification [11, 25] for example. One of the more interesting open issues concerns the extension of this method to the solution of Bilinear Matrix Inequalities [26, 27]. BMI's have recently been used to formulate controller synthesis problems in which controller order is explicitly constrained. The extension of our method to this class of problems would be particularly useful in that it would provide an efficient method for computing H^∞ controllers that explicitly bypasses the model reduction step usually required.

References

[1] C. C. Gonzaga, "Path-following methods for linear programming," *SIAM Review*, vol. 34, pp. 167–224, June 1992.

[2] M. H. Wright, "Interior methods for constrained optimization," *Acta Numerica*, pp. 341–407, 1991.

[3] M. S. Bazaraa, H. D. Sherali, and C. Shetty, *Nonlinear Programming: Theory and Algorithms*. New York: John Wiley and Sons, Inc., 1993.

[4] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, pp. 79–87, 1991.

[5] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the EM algorithm," Tech. Rep. 9301, MIT Computational Cognitive Science, Apr. 1993.

[6] M. I. Jordan and L. Xu, "Convergence results for the EM approach to mixtures of experts architectures," Tech. Rep. 9303, MIT Computational Cognitive Science, Sept. 1993.

[7] C. M. Bishop, "Mixture density networks," Tech. Rep. NCRG/4288, Dept. of Computer Science, Aston University, Birmingham, U.K., Feb. 1994.

[8] W. Byrne, "Alternating minimization and Boltzmann machine learning," *IEEE Transactions on Neural Networks*, vol. 3, pp. 612–620, July 1992.

[9] L.-X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Transactions on Neural Networks*, vol. 3, pp. 807–814, Sept. 1992.

[10] J. R. Koza, *Genetic Programming: on the programming of computers by means of natural selection.* Cambridge, Massachusetts: MIT Press, 1992.

[11] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach.* Boston, Massachusetts: Kluwer Academic Publishers, 1994.

[12] G. N. Saridis, "Entropy formulation of optimal and adaptive control," *IEEE Transactions on Automatic Control*, vol. 33, pp. 713–721, Aug. 1988.

[13] H. A. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. AC-33, pp. 780–783, Aug. 1988.

[14] Z. Ghahramani, "Solving inverse problems using an EM approach to density estimation," in *Proceedings of the 1993 Connectionist Models Summer School*, (Hillsdale, New Jersey), pp. 316–323, Erlbaum Associates., 1994.

[15] T. W. Cacciatore and S. J. Nowlan, "Mixtures of controllers for jump linear and nonlinear plants." Unpublished Paper.

[16] A. Dempster, N. Laird, and D.B.Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, pp. 1–38, 1977.

[17] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, vol. 26, pp. 195–239, Apr. 1984.

[18] R. A. Boyles, "On the convergence of the EM algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 45, no. 1, pp. 47–50, 1983.

[19] C. J. Wu, "On the convergence properties of the EM algorithm," *The Annals of Statistics*, vol. 11, no. 1, pp. 95–103, 1983.

[20] E. Singer and R. P. Lippmann, "Improved hidden Markov model speech recognition using radial basis function networks," in *Advances in Neural Information Processing Systems 4* (J. Moody, S. Hanson, and R. Lippmann, eds.), pp. 159–166, San Mateo, California: Morgan Kaufmann, 1992.

[21] S. J. Nowlan, "Maximum likelihood competitive learning," in *Advances in Neural Information Processing Systems 2*, pp. 574–582, San Mateo, California: Morgan Kaufmann Publishers, Inc., 1990.

[22] P. T. Szymanski, M. Lemmon, and C. J. Bett, "Training mixture radial basis function networks using an hybrid interior point algorithm." Submitted to *Neural Networks*, Aug. 1995.

[23] P. T. Szymanski, M. Lemmon, and C. J. Bett, "An hybrid interior point algorithm for performing alternating minimizations." Submitted to the *SIAM Journal of Optimization*, Sept. 1995.

[24] A. Nádas, D. Nahamoo, and M. A. Picheny, "On a model-robust training method for speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, pp. 1432–1436, 1988.

[25] J. Yang, Y. Xu, and C. Chen, "Hidden Markov model-based learning controller," in *Proceedings of the 1994 IEEE International Symposium on Intelligent Control*, (Columbus, Ohio), pp. 39–44, Aug. 1994.

[26] K. Goh, M. Safanov, and G. Papavassilopoulos, "A global optimization approach for the BMI problem," in *Proceedings of the* 33^{rd} *Conference on Decision and Control*, (Lake Buena Vista, Florida), pp. 2009–2014, 1994.

[27] K. Goh, J. Ly, L. Turan, and M. Safanov, "μ/k_m-synthesis via bilinear matrix inequalities," in *Proceedings of the* 33^{rd} *Conference on Decision and Control*, (Lake Buena Vista, Florida), pp. 2032–2037, 1994.