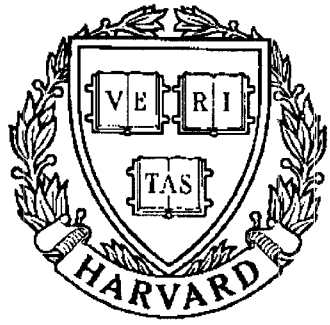
TECHNICAL RESEARCH REPORT

SYSTEMS RESEARCH CENTER



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

Online Parameter Optimization for a Multi-Product, Multi-Machine Manufacturing System

by J.S. Dhingra and G.L. Blankenship

TR 92-35

Online Parameter Optimization for a Multi-Product, Multi-Machine Manufacturing System*

Jastej S. Dhingra and Gilmer L. Blankenship

Systems Research Center
and
Department of Electrical Engineering
University of Maryland

Abstract

We develop an algorithm based on Infinitesimal Perturbation Analysis for online optimization of a multi-product service facility composed of a network of multi-server machines, modeled using multi-class M/M/m queues. Starting from the Robbins-Monro stochastic approximation method, we first develop an online, local optimization algorithm for a single multi-server machine. For the special case of Poisson arrivals and exponentially distributed service times in a multi-machine network, local optimization at individual machines leads to global optimization of the overall network. Simulation results for a single machine are compared to the exact analytical results. Application of the methodology for optimization of a simple flexible manufacturing system is also presented.

1 Introduction

In this paper we consider the problem of online control of a flexible manufacturing system.[1] A flexible manufacturing system is composed of a collection of multi-server machines, each capable of performing a variety of operations on different sets of parts. The parts are processed in a predefined operation sequence at different machines. The processing at each machine can be either batch or continuous mode. Batch mode operation involves processing a single part type over a given time interval (determined by the batch size), and then switching to process another part type. Batch mode processing is used if there is a significant setup time involved for the machine to switch from one part type processing to another part type processing. Continuous mode processing is used in cases where the

*This work was supported by NSF Engineering Research Centers Program NSFD CDR 88003012

[1] Although we treat manufacturing systems, it should become apparent that the methodology is generic and could be applied in many other cases involving control of multicommodity flows.

processing for different part types is very similar and changeovers do not require a significant reconfiguration of the machine setup.

In certain cases (where sets of physical machines can be grouped together to form "virtual machines" which process parts in tandem), manufacturing systems can also function in a hybrid mode, i.e., use batch mode processing for inter-virtual-machine operation and continuous mode processing (single product) within each virtual-machine. Since manufacturing systems typically exhibit variable production rates, random machine failures, unpredictable part availability and inter-machine blockage, performance optimization involves process models which are intrinsically stochastic.

In this paper we consider the problem of online control of the mean processing times for different parts at different machines to maximize the operational performance, subject to stability constraints, of a multi-machine manufacturing system operating under uncertainty. We use the cumulative system time as the performance measure (cost function).

Our formulation of the optimization problem is partitioned into two sub-problems: (i) estimation of the derivative of the performance measure with respect to the mean processing times; and (ii) use of gradient approximation methods for finding the optimal values for the parameters. To estimate the gradient of the performance measure, we use *Perturbation Analysis* (PA) method. PA was developed for sensitivity analysis in linear continuous time systems. Ho, Cao, Gong and Suri [8, 9, 4, 6, 7, 13, 15] have extended the basic ideas and applied them to discrete event dynamic systems.

The basic idea in perturbation analysis is to estimate the gradient of a given objective function, with respect to a parameter, for a stochastic dynamical system using a *single sample path.* Although this involves extra computational overhead (in simulations), it is more efficient than other finite difference based estimation techniques which involve separate sample path generation for several different values of the design parameter. PA has the added advantage that it permits gradient estimation with respect to several parameters using the same sample path. Suri and Leung [14] have used perturbation analysis for gradient estimation with respect to the processing and arrival rates for a single class G/G/1 queue. Fu and Hu [5] later extended the results to the single class GI/G/m queue and proved consistency of the gradient estimates. A contribution of our work is the optimization of multi-class queuing models.

The second step in the optimization is to use the gradient estimates for in stochastic optimization procedures. Two classical procedures used for stochastic optimization are the stochastic gradient, Robbins-Monro method, and the finite difference gradient estimation, Kiefer-Wolfowitz method. The Kiefer-Wolfowitz [10] procedure relies on gradient estimation using finite differences between a pair of sample paths, each generated using a different value of the design parameter. This involves dual sample path generation. Multiple parameter optimization requires separate dual path generation for each parameter. The Robbins-Monro [11] procedure, like perturbation analysis, uses a single sample path for stochastic optimization. The Robbins-Monro algorithm typically converges faster than the

Keifer-Wolfowitz method. Hence, we use it in this work.

In the next section, we develop the notation for the queuing system model and present an algorithm for derivative estimation using PA. We derive the optimization procedure based on the Robbins-Monro stochastic gradient algorithm in Section III. In Section IV, we present details of the implementation, typical simulation results, and comparisons with analytical estimates. The validity of the scheme is also discussed in Section IV.

2 Gradient Estimation Using Perturbation Analysis

If we consider a multiple machine manufacturing system with continuous mode processing, Poisson part arrivals, and exponentially distributed processing times, then global optimization of the system (in our formulation) can be reduced to a set of local optimization problems based on Burke's theorem [3]:

Theorem 1 (Burke) *The steady-state output of a queue with N channels in parallel, with Poisson arrival statistics, and with lengths chosen independently from an exponential distribution is itself Poisson.*

Therefore, if the part arrival in the manufacturing system is Poisson, all constituent machines will also have Poisson part arrivals. If they also have exponentially distributed processing times, local optimization at the machine level will lead to a global optimum for the whole manufacturing system. Hence, we can focus on optimization of the mean processing time for an M/M/m queue (multi-class case).

For each machine we define:

m	=	Number of servers in the machine.
K	=	Number of different customer types the machine is capable of processing.
λ_i	=	The arrival rates for different customer types $(i = 1 \ldots K)$
θ_i	=	The mean processing times for different customer types $(i = 1 \ldots K)$
ρ_i	=	Machine workload due to customer type i. $(= \frac{\lambda_i \theta_i}{m})$
λ	=	The net arrival rate of customers at the machine. $(= \sum_{i=1}^{K} \lambda_i)$
θ	=	The average processing time for customers at the machine. $(= \frac{\sum_{i=1}^{K} \lambda_i \theta_i}{\lambda})$
ρ	=	$\frac{\lambda \theta}{m} = \sum_{i=1}^{K} \rho_i$ = average workload on the machine.

The objective function C used in our study is the total time through the system. It is calculated as the sum of the respective cost functions C_i for each part type being processed at the machine $(C = \sum_{i=1}^{K} C_i)$. The individual cost components are defined in terms of the *average system time* T and the *mean processing time* θ_i. The optimization problem is:

$$\min_{0<\lambda\theta<m} \sum_{i=1}^{K} C_i(\theta_i, T)$$

To apply (stochastic) gradient optimization procedures, we need the gradient of C with respect to the individual mean processing times. The system time T can be written as the sum of actual processing time X and the queue delay D. Computation of the cost derivative can be reduced further using:

$$\begin{aligned}
\frac{dC}{d\theta_r} &= \sum_{i=1}^{K} \frac{dC_i(\theta_i, X, D)}{d\theta_r} \\
&= \sum_{\substack{i=1 \\ i\neq r}}^{K} \frac{dC_i(\theta_i, X, D)}{d\theta_r} + \frac{dC_r(\theta_r, X, D)}{d\theta_r} \\
&= \sum_{\substack{i=1 \\ i\neq r}}^{K} \frac{\partial C_i(\theta_i, X, D)}{\partial D}\frac{dD}{d\theta_r} + \frac{\partial C_r(\theta_r, X, D)}{\partial \theta_r} + \frac{\partial C_r(\theta_r, X, D)}{\partial X}\frac{dX}{d\theta_r} + \frac{\partial C_r(\theta_r, X, D)}{\partial D}\frac{dD}{d\theta_r} \\
&\qquad\qquad (\text{ As } \frac{d\theta_i}{d\theta_r} = \frac{dX}{d\theta_r} = 0 \quad \forall i \neq r) \\
&= \sum_{i=1}^{K} \frac{\partial C_i(\theta_i, X, D)}{\partial D}\frac{dD}{d\theta_r} + \frac{\partial C_r(\theta_r, X, D)}{\partial \theta_r} + \frac{\partial C_r(\theta_r, X, D)}{\partial X}\frac{dX}{d\theta_r}
\end{aligned}$$

The term $dX/d\theta$ represents the change in the actual processing time X due to the change in the mean processing time θ. For a smooth distribution function $F_\theta(X)$ for the processing time X, the derivative can be computed using the relation

$$\frac{dX}{d\theta} = -\left(\frac{\partial F_\theta(X)}{\partial \theta}\right) / \left(\frac{\partial F_\theta(X)}{\partial X}\right)$$

For the case of exponentially distributed service times, X with mean θ, the derivative is X/θ.

To understand the algorithm for derivative estimation, consider the simpler case of estimating $dD/d\theta_r$. Suppose we have a single server with multi-part arrivals. Assume that we only perturb the service parameter of a single part type (say part type 1). Under normal conditions, for the n^{th} part arrival in a specific busy period, let D_n and X_n be the queue delay and the processing time respectively. The new queue delay for the perturbed queue, i.e., with $\theta_1 \leftarrow \theta_1 + \Delta\theta_1$, will be D_n plus the sum of the change in the processing times for all the type 1 parts processed prior to the present part arrival, in the current busy period. As discussed earlier, for each type 1 part, the change in the processing time, due to a change

in the parameter, is given by $(dX/d\theta_1)\,\Delta\theta_1$. If we define $type(i)$ as the type of the i^{th} part, the change in the queue delay for the n^{th} customer is:

$$\Delta D_n = \sum_{\substack{i=1 \\ type(i)=1}}^{n-1} \left.\frac{dX}{d\theta_1}\right|_{X_i} \Delta\theta_1$$

If there are N part arrivals in an iteration interval (different from a busy period), the derivative $dD/d\theta_1$ can be estimated using

$$\frac{dD}{d\theta_1} \simeq \frac{1}{N}\sum_{n=1}^{N}\frac{\Delta D_n}{\Delta\theta_1} = \frac{1}{N}\sum_{n=1}^{N}\sum_{\substack{i=1 \\ type(i)=1}}^{n-1}\left.\frac{dX}{d\theta_1}\right|_{X_i}$$

If we change the processing parameter for more than one part type (say for all part types), then the respective partial derivatives can be computed using

$$\frac{\partial D}{\partial\theta_r} \simeq \frac{1}{N}\sum_{n=1}^{N}\frac{\Delta D_n(r)}{\Delta\theta_r} = \frac{1}{N}\sum_{n=1}^{N}\sum_{\substack{i=1 \\ type(i)=r}}^{n-1}\left.\frac{dX}{d\theta_r}\right|_{X_i} \qquad r = 1\ldots K$$

Here $\Delta D_n(r)$ is the partial contribution in the net change in queue delay due to the change in θ_r. This analysis is for a single server. For m servers, the net delay change tracking is done separately for all servers and the derivative estimation is done over the whole m server queue

$$\frac{\partial D}{\partial\theta_r} \simeq \frac{1}{N}\sum_{n=1}^{N}\frac{\Delta D_n^s(r)}{\Delta\theta_r} = \frac{1}{N}\sum_{n=1}^{N}\sum_{\substack{i=1 \\ type(i)=r \\ server(i)=s=server(n)}}^{n-1}\left.\frac{dX}{d\theta_r}\right|_{X_i} \qquad r = 1\ldots K$$

Here $server(i)$ represents the server-id of the server processing the i^{th} part and N is the total number of parts processed by the m servers in the current iteration period.

Our problem is to estimate the cost derivatives $dC/d\theta_r$. For each part arrival in an iteration period, we calculate the change in the cost and take the ensemble average at the end of the iteration period. For the n^{th} arrival of type r, with X_n and D_n as the processing and the queue delay respectively, the overall cost change is given by

$$\Delta C_n = \sum_{i=1}^{K}\left.\frac{\partial C_i(\theta_i, X, D)}{\partial D}\right|_{X_n,D_n}\Delta D_n^s(r) + \left.\frac{\partial C_r(\theta_r, X, D)}{\partial\theta_r}\right|_{X_n,D_n}\Delta\theta_r$$

$$+ \left.\frac{\partial C_r(\theta_r, X, D)}{\partial X_n}\right|_{X_n, D_n} \left.\frac{dX}{d\theta_r}\right|_{X_n} \Delta\theta_r$$

This net cost change is split into the component changes $\Delta C_n^r \quad r = 1 \ldots K$, each of which can be attributed to the respective changes in the individual processing parameters (θ_r). At the end of the iteration period, the average cost change is used as the derivative estimated.

Formalizing the procedure as an algorithm, we have:

INITIALIZE:

for ($server = 1 \ldots m$) do
  for ($type = 1 \ldots K$) do
    ($\Delta D^{server}(type) \leftarrow 0$)

for ($type = 1 \ldots K$) do
  ($\Delta C(type) \leftarrow 0; \Sigma_{type} \leftarrow 0$)

AT COMPLETION OF PART(n) PROCESSING:

($s \leftarrow server(n); r \leftarrow type(n)$)

($\Sigma_r \leftarrow \Sigma_r + 1$)

for ($type = 1 \ldots K$) do
  ($\Delta C(type) \leftarrow \Delta C(type) + \left.\frac{dC_{type}}{dD}\right|_{X_n, D_n} \Delta D^s(r)$)

($\Delta C(r) \leftarrow \Delta C(r) + \left.\frac{dC_r}{d\theta_r}\right|_{X_n, D_n} + \left.\frac{dC_r}{dX}\right|_{X_n, D_n} \left.\frac{dX}{d\theta_r}\right|_{X_n}$)

($\Delta D^s(r) \leftarrow \Delta D^s(r) + \left.\frac{dX}{d\theta_r}\right|_{X_n}$)

if (s idle) then
  for ($type = 1 \ldots K$) do
    ($\Delta D^s(type) \leftarrow 0$)

AFTER N COMPLETIONS:

for ($type = 1 \ldots K$) do
  ($\frac{dC}{d\theta_{type}} \simeq \frac{1}{\Sigma_{type}} \Delta C(type)$)

3 Optimization Based on Stochastic Approximation

The Robbins-Monro procedure was developed initially as a root finding method for a random function of a single variable. This procedure can be used as an stochastic optimization method by applying it to find the root of the gradient function. The original Robbins-Monro algorithm, as extended to the multi-dimensional case by Blum [2], permits treatment of multi-dimensional stochastic optimization problems.

For a given cost functional $J(Y)$, the optimization problem reduces to finding the root of $\nabla_Y J(Y^*) = 0$. Here $Y = \{y_1, \ldots y_k\}$ is the vector of random parameters, over which the optimization is carried out. The optimization is done by recursive reassignment of the control parameter vector Y to the series of random vectors $\{Y_n\}_{n=0}^{\infty}$. The series $\{Y_n\}_{n=0}^{\infty}$ is generated using the recursive relation

$$Y^{n+1} = Y^n + A_n \widehat{\nabla}_Y J(Y^n)$$

. Here $\widehat{\nabla}_Y J(Y^n)$ is the estimate of the noisy gradient and $A_n = \mathrm{diag}[a_n^1, \ldots a_n^k]$.

Blum [1, 2] proved that if we define the sequences $\{a_n^i\}_{n=0}^{\infty}$, $i = 1 \ldots k$ such that

$$\begin{aligned} \lim_{n\to\infty} a_n^i &= 0 && \forall i \\ \sum_{n=1}^{\infty} a_n^i &= \infty && \forall i \\ \sum_{n=1}^{\infty} (a_n^i)^2 &< \infty && \forall i \end{aligned}$$

and $\widehat{\nabla}_Y J(Y^n)$ is an unbiased estimate of $\nabla_Y J(Y)$, then the sequence $\{Y^n\}_{n=0}^{\infty}$ converges *a.s.* to Y^*.

4 Implementation and Results

To measure the performance of a multi-machine facility, we use component cost functions of the form:

$$C_i(\theta_i, X, D) = a_i(X + D) + \frac{b_i m}{\theta_i}$$

For a fixed number of servers (m) the cost function is convex, and for $0 < \rho < 1$, it has a unique minimum. The first term is proportional to the average system time $(X + D)$

while the second term is proportional to the speed of each server $(1/\theta_i)$. The tradeoff between costly, but faster servers, and longer system times creates a unique minimum for appropriate choices of the coefficients a_i and b_i. The average queue delay D is itself a function of $\{\theta_i\}_{i=1}^K$. The gradients can be computed for this cost functional by solving the following set of equations:

$$\frac{d\mathrm{E}[C]}{d\theta_r} = 0 = \sum_{i=1}^{K} a_i \frac{d\mathrm{E}[D]}{d\theta_r} + a_r \frac{d\mathrm{E}[X]}{d\theta_r} - \frac{b_r m}{\theta_r^2} \qquad r = 1 \ldots K$$

substitutingE$[X] = \theta_r$, we have

$$\frac{d\mathrm{E}[C]}{d\theta_r} = 0 = \sum_{i=1}^{K} a_i \frac{d\mathrm{E}[D]}{d\theta_r} + a_r - \frac{b_r m}{\theta_r^2} \qquad r = 1 \ldots K$$

For the multiple class, multi-server case we can define

$$\lambda = \sum_{i=1}^{K} \lambda_i, \qquad \theta = \frac{1}{\lambda} \sum_{i=1}^{K} \lambda_i \theta_i \qquad \text{and} \qquad \rho = \frac{1}{m} \sum_{i=1}^{K} \lambda_i \theta_i$$

From queuing theory, for a m server queue with intensity ρ and total arrival rate λ, the average queue delay is given by

$$\mathrm{E}[D] = P_0 \frac{(m\rho)^m}{m!(1-\rho)} \frac{\rho}{\lambda(1-\rho)}$$

where

$$P_0 = \left[\sum_{i=0}^{m-1} \frac{(m\rho)^i}{i!} + \frac{(m\rho)^m}{m!(1-\rho)} \right]^{-1}$$

The optimization problem can be reduced to solving the following set of equations for $\{\theta_r\}_{r=1}^K$

$$\frac{d\mathrm{E}[C]}{d\theta_r} = 0 = \sum_{i=1}^{K} a_i \frac{d\mathrm{E}[D]}{d\theta_r} + a_r - \frac{b_r m}{\theta_r^2} \qquad r = 1 \ldots K$$

$$\iff \qquad \frac{1}{\lambda_r} \left[\frac{b_r m}{\theta_r^2} - a_r \right] = constant = \frac{1}{\lambda} \sum_{i=1}^{K} a_i \frac{d\mathrm{E}[D]}{d\theta} \qquad r = 1 \ldots K$$

These equations can be readily solved for the theoretical optimal values. To test the on-line algorithm, a single m-server queue, with two different part types was simulated using QNAP2 (Queuing Network Analysis Package). Figures 2 and 3 represent online optimization for the two cases of $m = 5$ and $m = 10$. In both cases we had $\lambda = 12$, the cost parameters $[a_1, a_2] = [20.0, 0.1]$ and $[b_1, b_2] = [1.0, 1.0]$. In both cases we started with $\lambda_1 = 8$ and $\lambda_2 = 4$. After each 25000 time units ($\Leftrightarrow$ 300,000 part arrivals $\Leftrightarrow$ 150 parameter updates) the arrival rates for the two types were interchanged. The solid lines represent the analytical, optimal values for the two mean processing times during each interval. Clearly, our scheme performs well, in the sense that the mean processing times converge quickly to the theoretical optimum values.

The validity of our scheme also follows from the results of De Smit [12], Fu and Hu [5] and Blum [1, 2]. Specifically, De Smit [12] proved that the multiple class, multi-server queue with Poisson arrivals and exponentially distributed times is equivalent to the GI/H_K/m queue. Here H_K denotes the hyperexponentially distributed service times, with distribution function of the form:

$$F_\theta(X) = \begin{cases} \sum_{i=1}^{K} \frac{\lambda_i}{\lambda}(1 - e^{-\frac{X}{\theta_i}}) & X \geq 0 \\ 0 & X < 0 \end{cases}$$

Fu and Hu [5] proved consistency of the gradient estimates for a GI/G/m queue. Those results are directly applicable to the GI/H_K/m queue; and so, the gradient estimates are consistent. By an appropriate choice of the sequence a_n^i for all i, our optimization parameters $\{\theta_i^n\}_{n=0}^{\infty}, (i = 1 \ldots K)$ will converge to the optimum $\theta_i^*(i = 1 \ldots K)$. For our simulation we chose $a_n^i = a_0^i/n, \forall i$ which satisfies all the stochastic approximation conditions discussed earlier.

We also applied the algorithm to online optimization of a small manufacturing system. Figure 1 shows a flow diagram for a four machine, two-product manufacturing system. This system was simulated using QNAP2; and online optimization was done over all machines. The simulation was started with $\lambda_1 = 8$ and $\lambda_2 = 4$. At time = 50,000 units, the two arrival rates were interchanged. Figures 4-6 show the online optimization for the four machines respectively. The mean processing time, for each part type, at each machine converges to the optimum value. This demonstrates global optimization of the system performance via local optimization at the machine level. Of course, this result holds only under the assumptions on Poisson arrivals that we set out earlier.

References

[1] J. R. Blum, "Approximation methods which converge with probability one," *Annals of Math. Stat.*, 25, pp. 382-386, 1954.

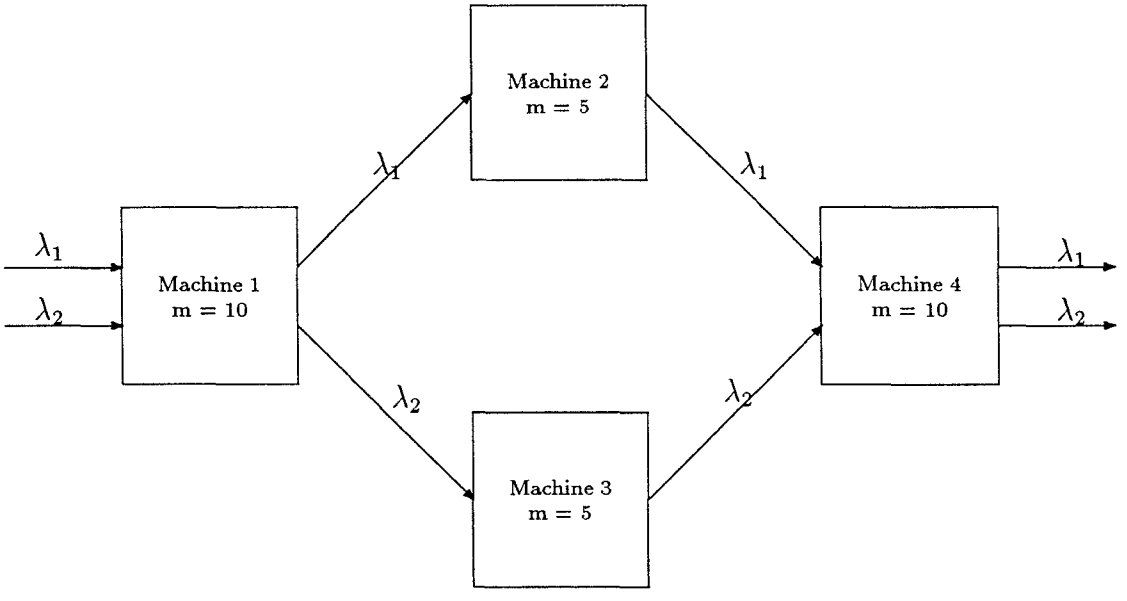


Figure 1: Manufacturing system flow

[2] J. R. Blum, "Multidimensional stochastic approximation methods," *Annals of Math. Stat.,* 25, pp. 737-744, 1954.

[3] P.J. Burke, "The output of a queuing system," *Operations Research,* 4, pp. 699-704, 1956.

[4] X. R. Cao and Y. C. Ho, "Sensitivity analysis and optimization of throughput in a production line with blocking," *IEEE Transactions on Automatic Control,* 32, pp. 959-967, 1987.

[5] M. C. Fu and Jian-Qiang Hu, "Consistency of infinitesimal perturbation analysis for the GI/G/m queue," *European Journal of Operational Research,* 54, pp. 121–139, 1991.

[6] W. B. Gong and Y. C. Ho, "Smoothed (conditional) perturbation analysis of discrete event dynamical systems," *IEEE Transactions on Automatic Control,* 32, pp. 858-866, 1987.

[7] W. B. Gong, "Smoothed perturbation analysis of Markovian queuing networks," *Proc. American Control Conference,* pp. 456-461, 1988.

[8] Y. C. Ho, "Performance evaluation and perturbation analysis of discrete event dynamic systems," *IEEE Transactions on Automatic Control,* 32, pp. 563-572, 1987.

[9] Y. C. Ho, "Extensions of infinitesimal perturbation analysis," *IEEE Transactions on Automatic Control,* 33, pp. 427-438, 1988.

[10] J. Keifer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *Annals of Math. Stat.,* 23, pp. 462-466, 1952.

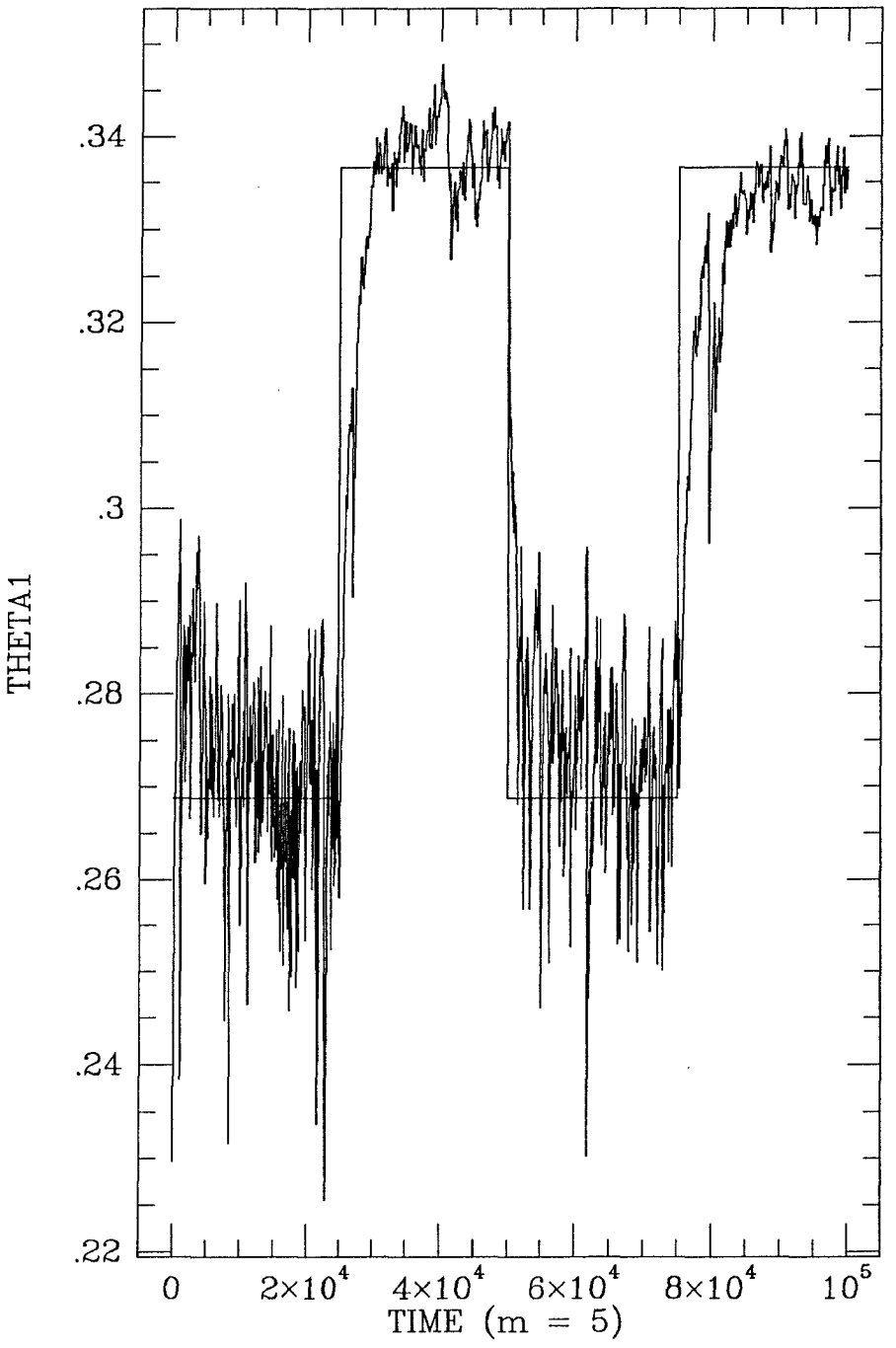
[11] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Math. Stat.,* 22, pp. 400-407, 1951.

[12] J. H. A. De Smit, "The queue GI/M/s with customers of different types or the queue GI/H_m/s," *Advances in Applied Probability,* 15, pp. 392-419, 1983.

[13] R. Suri, "Infinitesimal perturbation analysis for general discrete event systems," *Journal of the ACM,* 34, pp. 686–717, 1987.

[14] R. Suri and Y. T. Leung, "Single run optimization of discrete event simulations - an empirical study using the M/M/1 queue," *IIE Transactions,* 21, pp. 35-48, 1989.

[15] R. Suri, "Perturbation analysis: The state of the art and research issues explained via the G/G/1 queue," *Proceedings of the IEEE,* 77, pp. 114–137, 1989.


THETA1
.34
.32
.3
.28
.26
.24
.22
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME (m = 5)

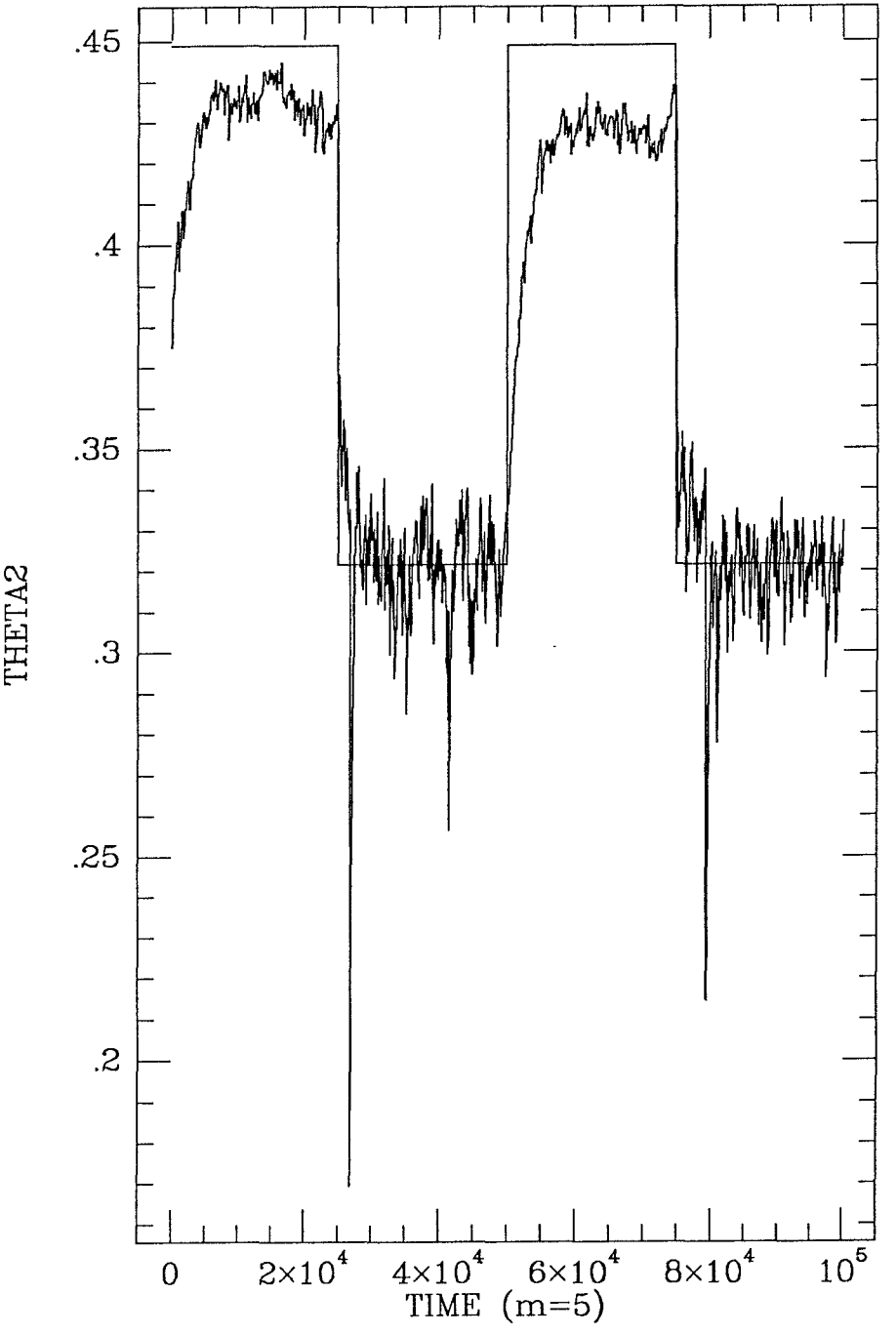
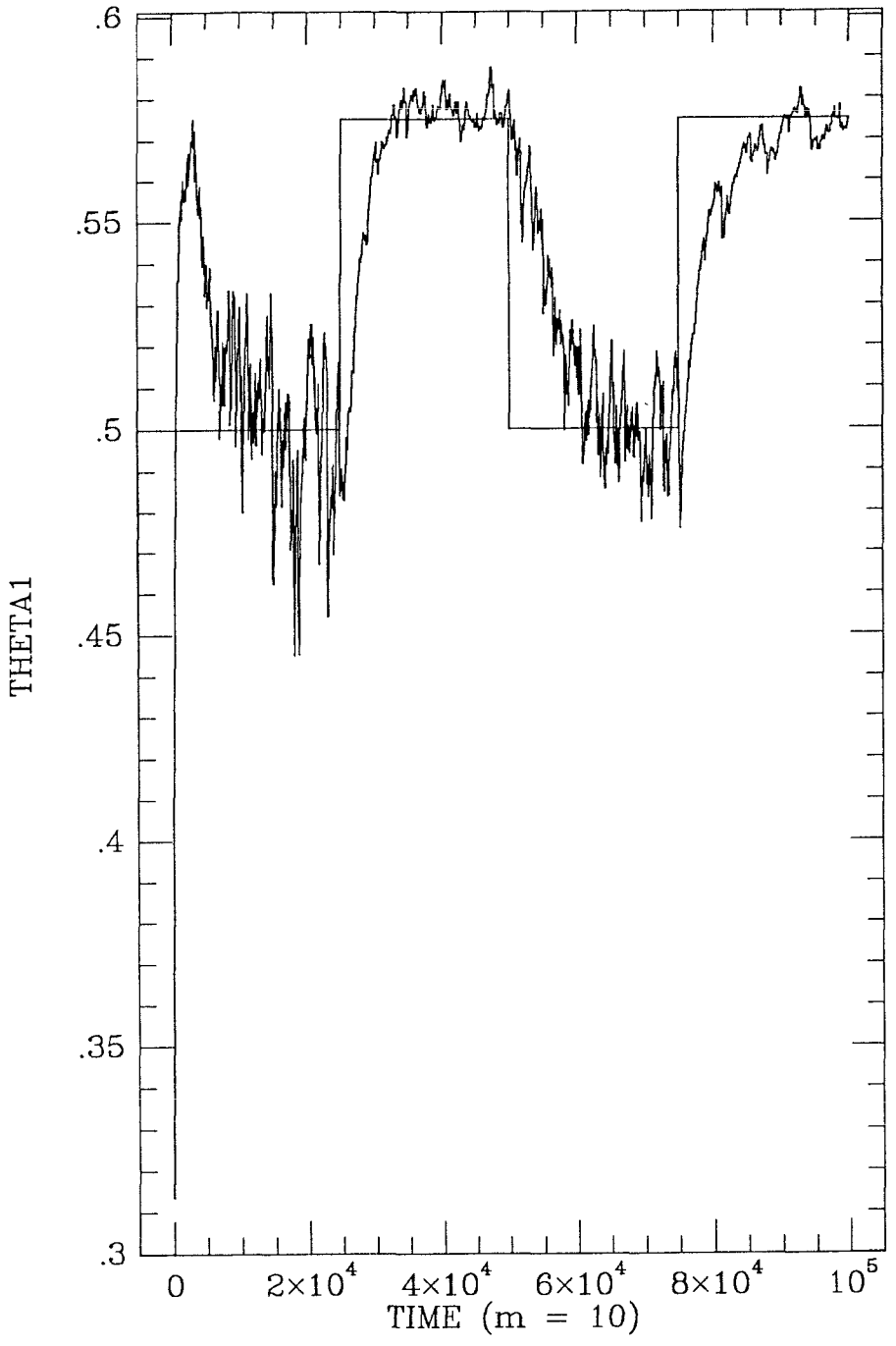

THETA2
.45
.4
.35
.3
.25
.2
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME (m=5)

FIGURE 2


.6
.55
.5
.45
.4
.35
.3
THETA1
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME (m = 10)

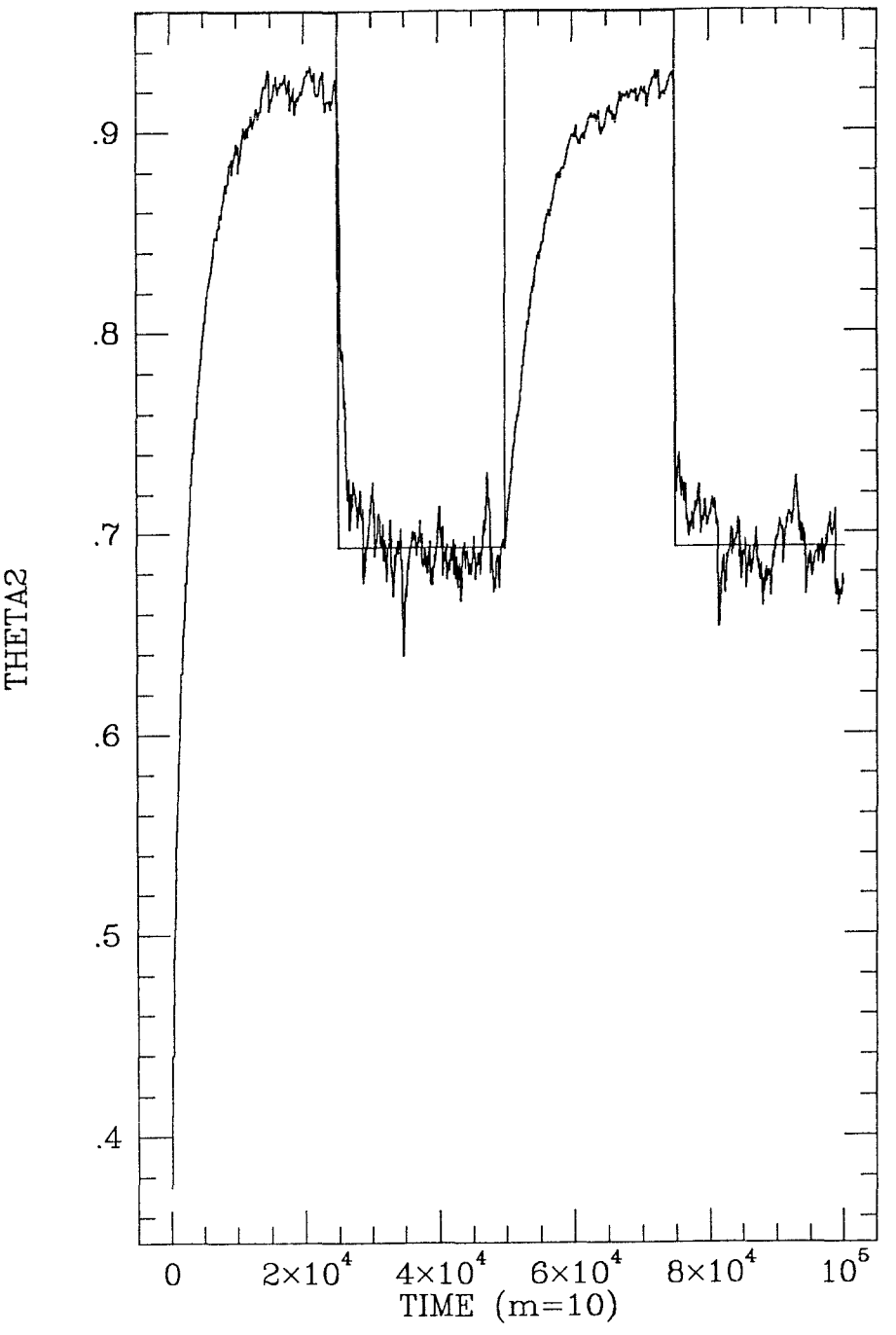
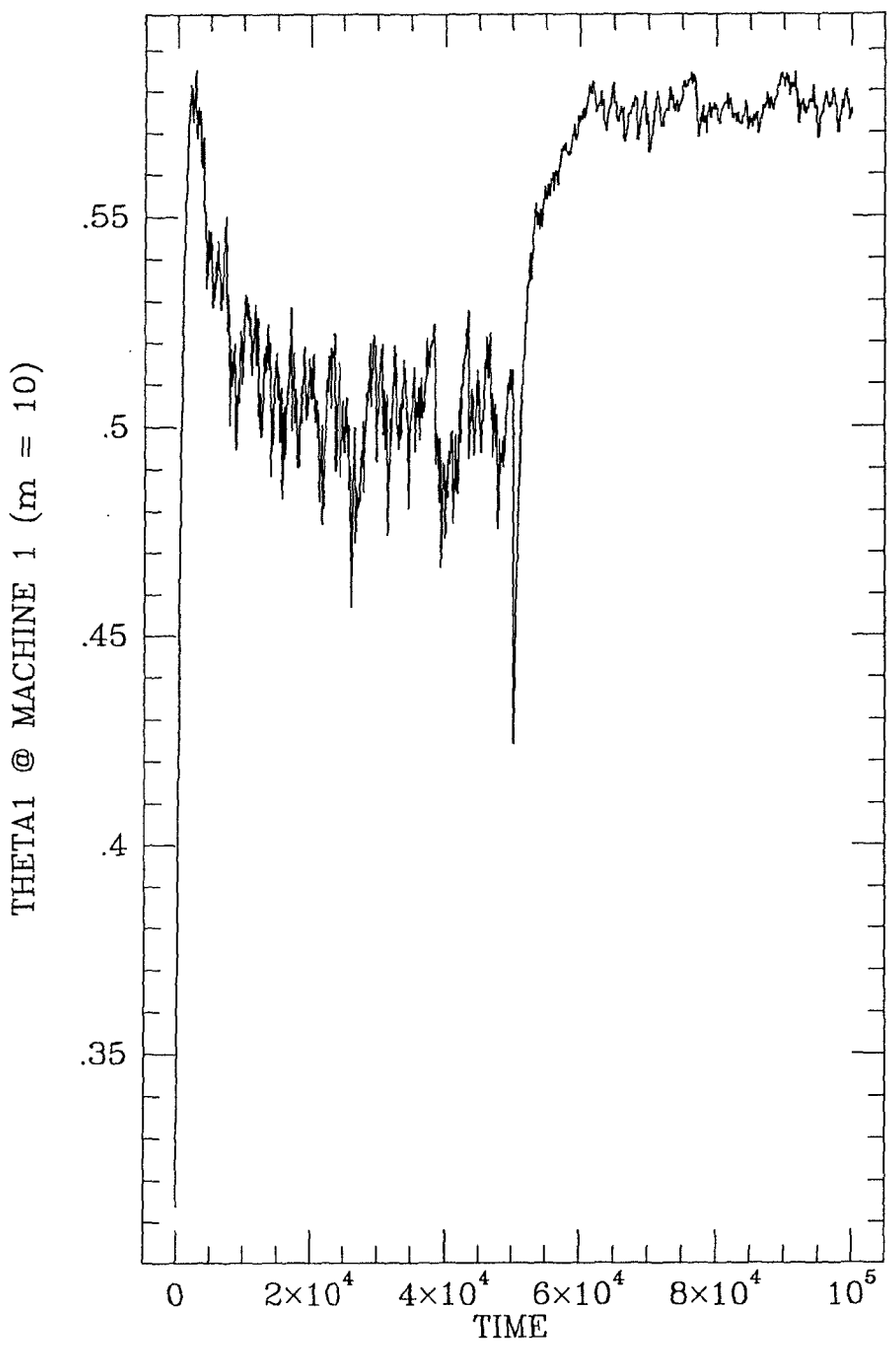

.9
.8
.7
.6
.5
.4
THETA2
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME (m=10)

FIGURE 3


THETA1 @ MACHINE 1 (m = 10)
.55
.5
.45
.4
.35
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME

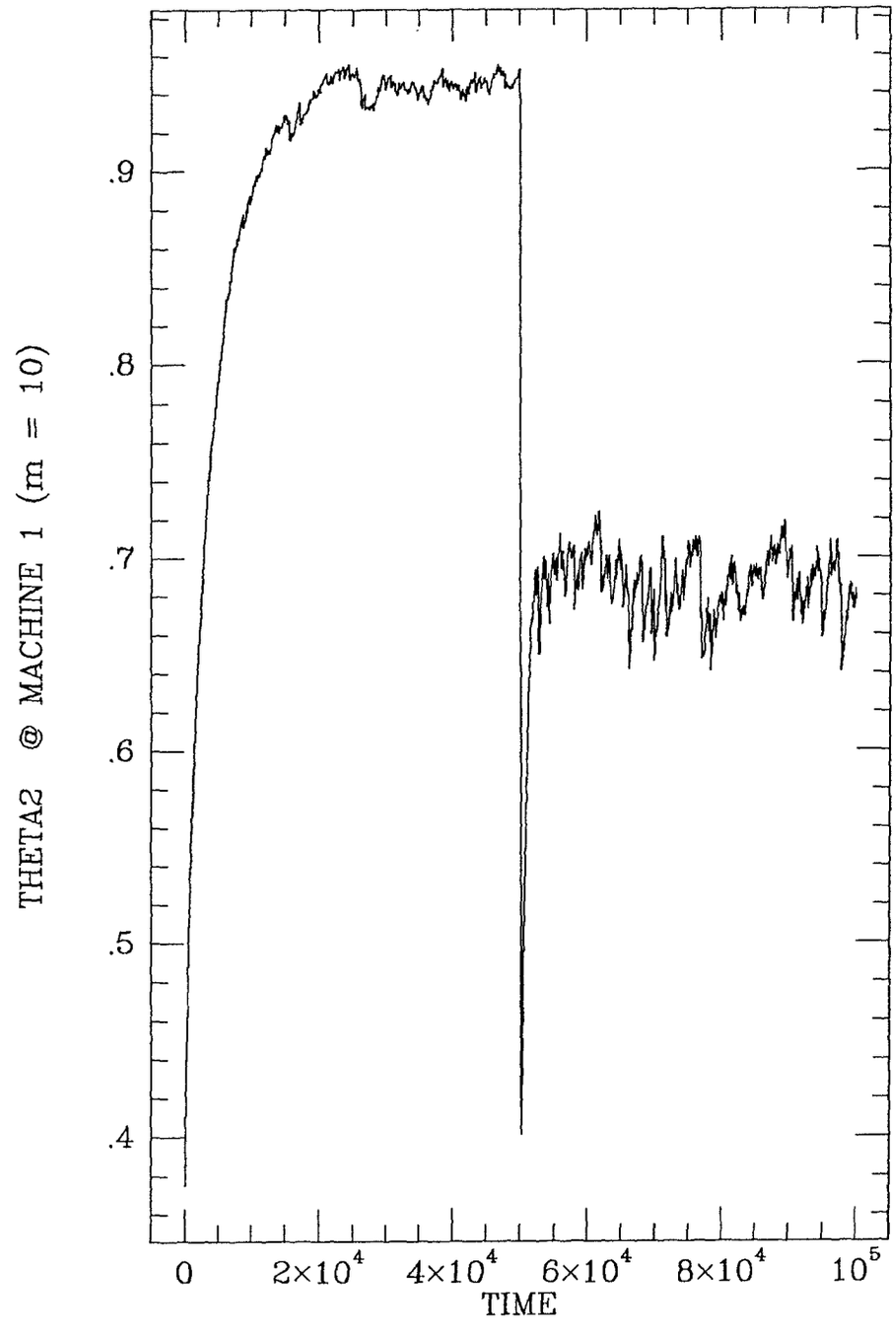
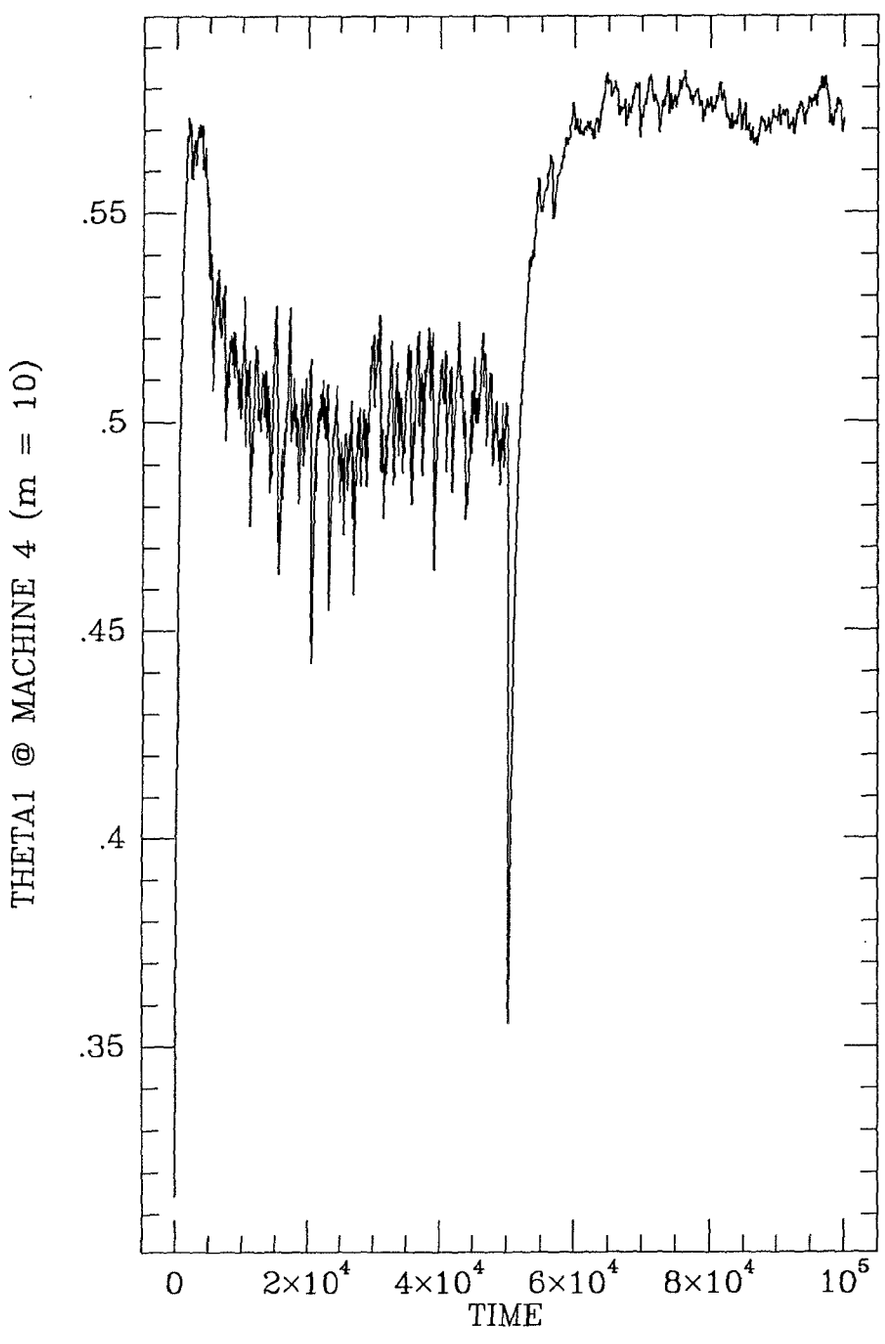

THETA2 @ MACHINE 1 (m = 10)
.9
.8
.7
.6
.5
.4
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME

FIGURE 4

THETA1 @ MACHINE 2 (m = 5)
.45
.4
.35
.3
.25
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME

THETA2 @ MACHINE 3 (m = 5)
.7
.65
.6
.55
.5
.45
.4
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME

FIGURE 5


THETA1 @ MACHINE 4 (m = 10)
.55
.5
.45
.4
.35
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME

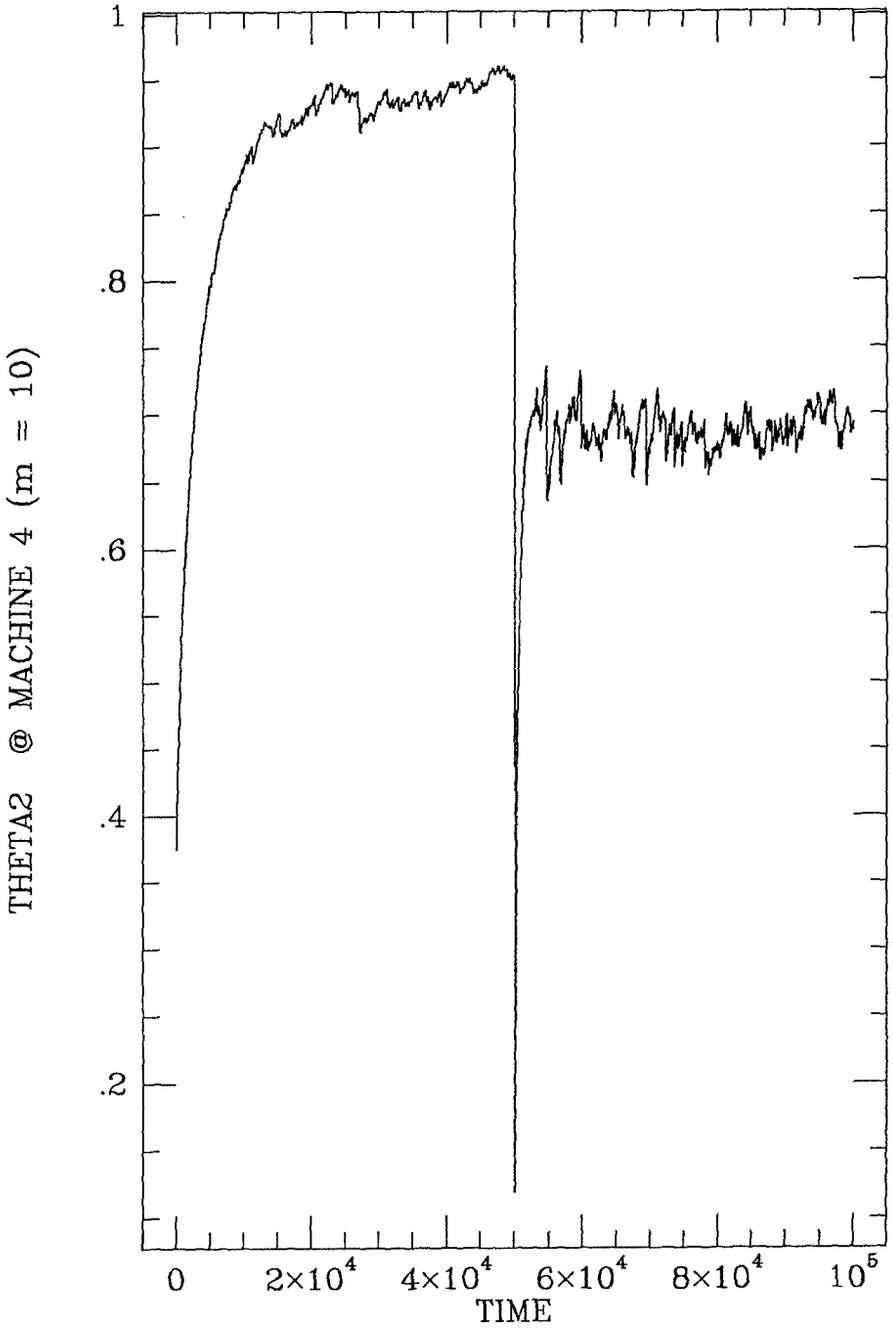

THETA2 @ MACHINE 4 (m = 10)
1
.8
.6
.4
.2
0
2×10^4
4×10^4
6×10^4
8×10^4
10^5
TIME

FIGURE 6