Optimization of Process Performance

LEON LAPIDUS, EUGENE SHAPIRO, SAUL SHAPIRO, and R. E. STILLMAN

IBM Research Center, Yorktown Heights, New York

The growing availability of fast, large memory digital computers has made it practical to consider the physical implementation of control system designs incorporating appropriate strategies for automatic process optimization. The control system is taken to consist of the process to be optimized together with the interconnected digital computer. The control strategy is realized in the program of the digital computer. In the present paper a number of such programs or algorithms are discussed for carrying out a search of the possible settings of the process input (independent) variables in such a way as to locate an extremal of the possible values of a chosen objective function. The magnitude of these variables is determined from measurements taken of the dependent variables in the process. It is shown that for the particular process used as an example it is desirable to alter the search strategy as the optimization proceeds in order to locate the extremal in a minimum amount of time. The emphasis at the beginning of the search is on speed in moving towards the optimum and at the end on accuracy.

Further, a computational technique is described whereby the dynamic response of the process to the various search steps (or settings) is under time-optimal control. This procedure is important for the fast execution of the search programs and consequent rapid location of the extremal of the chosen objective function.

THE SEARCH PROBLEM

A typical optimization problem often encountered in chemical engineering can be expressed by the following statement. Given a functional relationship

$$y^* = f(x_1, x_2, ----, x_n)$$

minimize (or maximize) the response function y^* by a systematic manipulation of the independent variables $x_1, x_2, ---, x_n$ which may or may not be constrained. The functional relationship $f(x_1, x_2, ---, x_n)$ is usually highly nonlinear with the exact form unknown. Note that since y^* is not an explicit function of time, the systematic manipulation of the independent variables does not require a knowledge of the system dynamics. Once the optimum settings of these variables are obtained, the optimization procedure is completed. However if the properties of the system tend to change with time, the entire procedure for finding these optimum settings must be repeated at periodic intervals. Thus the so-called "static" or "single-stage optimization procedure" is modified by repeated application with the end result being, in effect, a dynamic or multiple-stage process.

There are a number of classic methods in existence with varying degrees of usefulness for the general solution of this type of problem. The one considered here is a gradient technique (method of steepest descent or ascent) in which the search is based on repeated explorations of a response surface. The information gained from these explorations is used to determine new optimal settings for the independent variables. Care must be exercised in the application of this technique, since it is only useful for finding a local minimum (or maximum). Thus an extensive exploration of the response surface may be required to determine whether a relative or an absolute minima (or maxima) has been found. The application of gradient techniques to the specific solution of process optimization problems is certainly not new (*6*). It has been shown however that in certain cases of practical interest use of a gradient technique by itself does not provide the most efficient algorithm (or procedure) for locating a process optimum (*8*). Instead it is necessary to adjust the algorithm as the search proceeds in such a manner as to carry out the search as rapidly as possible when one is far away from the optimum, and as accurately as possible when one is close to the optimum. Since meeting these two requirements for any single process can result in the need to modify the classical gradient procedures, the authors first review in brief the essential characteristics of the pure or unmodified gradient technique (*7*).

The gradient method can be expressed by a generalized equation of the form

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) - \lambda(t_k)\,\mathbf{G}(t_k)$$

This equation represents a simple iterative procedure in which the new optimal settings can be obtained for the $\mathbf{x}$ vector at time $k+1$ directly from the values of $\mathbf{x}$, $\mathbf{G}$, and λ at time k. Thus the gradient method for the step-by-step approach to an optimum point may be divided into two parts: first, finding the direction $\mathbf{G}$ in which the function will decrease (or increase) a maximum amount per step taken, and second, finding the step size λ which will bring the function nearest to its minimum (or maximum) by stepping in that direction previously decided upon.

In the optimization of the y^* function, the change in y^* with respect to a given independent variable x_i, represents the partial derivative of y^* with respect to that variable and may be denoted by C_i. It can be shown that the change in y^* per length of function arc is greatest if the step sizes Δx_i are proportional to the C_i (*2*). Although the step size is not fixed, the change in each independent variable must be proportional to the partial derivatives at the point for maximum change in y^*. The determination of these partial derivatives is then the key factor in deciding the direction of the step. Three techniques will be described here.

Calculation of the Partial Derivatives by Finite Differences

This method involves the direct calculation of the partial derivatives. If each of the independent variables is perturbed successively from a given initial base point by some arbitrarily small amount and the corresponding responses measured, the differences between these measured responses and the response at the base point divided by the corresponding perturbation distance would represent crude estimates of the partial derivatives. The advantage of this method is that it allows one to use the system itself as a model. The disadvantages are that it requires $n + 1$ experiments for each point being explored, and it also does not provide a simple means for taking advantage of the past experience gained in exploring the response surface. Another disadvantage of this technique is that a single perturbation of each variable is only one direction results in a poorer estimate of the partial derivatives than other methods with more experiments.

Calculation of the Partial Derivatives by a Factorial Design Experiment

A much better estimate of the partial derivatives may be obtained by perturbing each independent variable in two directions, calculating the change in the dependent variable for each perturbation of the remaining independent variables, and averaging these changes and dividing by the Δx_i. This technique is known as a *factorial design experiment*. For example to evaluate $\partial y^*/\partial x_1$ with three independent variables x_1, x_2, x_3 it is necessary to evaluate

$$\frac{1}{8\Delta x_1}\left\{\begin{array}{l} \quad y^*(x_1 + \Delta x_1, x_2 + \Delta x_2, x_3 + \Delta x_3) - y^*(x_1 - \Delta x_1, x_2 + \Delta x_2, x_3 + \Delta x_3) \\ + y^*(\ +\ ,\ +\ ,\ -\ ) - y^*(\ -\ ,\ +\ ,\ -\ ) \\ + y^*(\ +\ ,\ -\ ,\ +\ ) - y^*(\ -\ ,\ -\ ,\ +\ ) \\ + y^*(\ +\ ,\ -\ ,\ -\ ) - y^*(\ -\ ,\ -\ ,\ -\ ) \end{array}\right\}$$

Leon Lapidus is at Princeton University, Princeton, New Jersey.

Geometrically the function is evaluated at eight points of a cube with the present position at its center. The partial derivative of any independent variable is obtained by subtracting function values at all edges of the cube parallel to the axis of that variable, averaging them, and dividing by the length of the edge. A disadvantage of this method is that it requires 2^n experiments to obtain the partial derivatives at a point. This disadvantage becomes even more pronounced as n increases.

Calculation of the Partial Derivatives by Polynomial Approximation

This method for finding the partial derivatives at a point uses a linear equation to approximate the function in the neighborhood of the point. The coefficients of each independent variable in the linear equation are estimates of the partial derivatives. The minimum number of experiments required to obtain this linear function is $n + 1$ (n independent variables), with this minimum case being equivalent to the finite difference technique. However the usual procedure is to use more than $n + 1$ experiments so that a least-squares fit may be made to obtain the required linear function. This method offers the advantage of allowing the use of a variable number of experiments for obtaining the partial derivatives. The use of the least-squares procedure for the estimation of the partial derivatives also allows one to retain the use of the past history of the response surface, and although the initial exploration requires at least $n + 1$ experiments, only one more experiment is required for each point considered thereafter. The disadvantage of this method is that the linear approximation used may not be an adequate representation of the response surface in the local region being explored.

The least-squares procedure may be summarized briefly as follows. A linear polynomial approximation to the local response surface is first written in the form

$$y(t_k) = C_o + C_1x_1(t_k) + C_2x_2(t_k) + \cdots + C_nx_n(t_k)$$

with $C_i = \partial y/\partial x_i$, $i = 1, 2, \cdots, n$. This may also be written in matrix form as

$$y(t_k) = \mathbf{x}(t_k)^T\,\mathbf{C}$$

where

$$\mathbf{x}(t_k) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}; \quad \mathbf{C} = \begin{bmatrix} C_o \\ C_1 \\ C_2 \\ \cdot \\ \cdot \\ \cdot \\ C_n \end{bmatrix}$$

The least-squares method is based upon the minimization of the sum of the squares of the residuals; that is

$$\text{minimize} \sum_i [y_i(t_k) - y_i^*(t_k)]^2$$

On this basis the least-squares solution for the partial derivatives is given by

$$\mathbf{C} = [\sum_{k=1}^{N} \alpha^{N-k}\,\mathbf{x}(t_k)\,\mathbf{x}(t_k)^T]^{-1}$$

$$[\sum_{k=1}^{N} \alpha^{N-k}\,\mathbf{x}(t_k)\,y^*(t_k)]$$

The weighting function $\alpha \leq 1$ was introduced to weight the most recent points in the response surface more heavily.

Formation of the Gradient Vector

Any of the above procedures can be used to estimate the partial derivatives needed for the formation of the **G** vector, that is the direction to move to decrease (or increase) the response function most rapidly. The simplest form for the gradient is to use

$$\mathbf{G}(t_k) = \mathbf{C}$$

that is, directly equal to the vector of the partial derivatives. However experience has shown that this form of the gradient can be greatly misleading particularly when dealing with nonlinear systems. A gradient utilizing some form of normalized partial derivatives such as

$$G_i(t_k) = \frac{C_i}{\left(\sum_{i=1}^{n} C_i^2\right)^{1/2}}$$

has proven to be more reliable. An even better form is the one suggested by Marquardt (*10*):

$$G_i(t_k) = \frac{(1 + [x_i(t_k)]^2)C_i}{\left\{\sum_{i=1}^{n} [(1 + [x_i(t_k)]^2)C_i]^2\right\}^{1/2}}$$

which represents a weighted—normalized gradient. This form of the gradient tends to tailor itself to the magnitudes of the variables. Its use reduces the overcorrecting of small valued variables and the undercorrecting of large valued variables.

Determination of the Step Size

Once the best direction has been found, the second part of the gradient procedure is to step in that direction. Proceeding in this direction one will bring the function toward the optimum point and then away from it (providing a finite optimum exists). The point at which the derivative changes sign (the point where the function value starts receding from the optimum) is the point closest to the optimum lying along this direction. The best strategy would be to step to this closest point in one step, calculate a new best direction, step again to the closest point in one step, calculate a new direction, and so on until the step size became zero and the optimum point is reached. The problem is thus to find a step size along the direction line which will bring the function to this closest point. In general the only information available at the time of the initial step is the first partial derivatives, and thus it is difficult to take a step which will arrive exactly at the closest point.

Hence in actual practice the determination of the step size remains somewhat arbitrary, the basic requirement being that it be of such a size so as to insure a reduction in the value of the response function for the newly calculated variables. When a step is taken, the new function value is tested, and if it is found to be further from the optimum, the step size is reduced in magnitude until a function value closer to the optimum is obtained. The question then arises should a second step be taken in that same direction or should a new direction be calculated. Although the desirability of always stepping in the best direction is obvious, it is noted that while it takes only one experiment per step (to compare the new function value with the old one), it takes from 1 to 2^n (depending on the method of finding the partial derivative) to calculate a new direction. Qualitatively it is seen that a good determination of the direction would enable stepping a greater distance along a calculated direction line, while a poorer determination of direction might be tolerated if a direction is calculated at each step. The least-squares procedure is an example of the latter technique, while the factorial-design method is an example of the former one.

In the least-squares procedure one very elementary way for determining the step size is to start with $\lambda(t_k) = 1$ and then to either double the value or divide it by four depending on whether the value of the newly obtained response function decreases or increases in any given step. This method has been found to give rapid convergence to a minimum independent of the starting point. A somewhat more sophisticated procedure for adjusting the step size is to use the angle between successive evaluations of the gradient vector to help serve as a guide as to when to increase or decrease the step size. Thus if the angle increases too much the step size is decreased, and if the angle decreases too much the step size is increased. The disadvantage with this last method is that it tends to give very slow convergence as one approaches close to a minimum. In fact it has been found for most cases that convergence can be accelerated near a minimum by switching from the least-squares procedure to a factorial-design technique.

The factorial-design method represents the other way of stepping; one steps in the calculated best direction until the function starts receding from the optimum and then calculates a new direction. In this case the step size may be a constant multiplied by the gradient or it may vary. It may be varied by extrapolating the partial derivatives of several points to the point where it is zero and stepping that distance. Another method is to start with a very small step and to double the step size until the function value starts receding in value and then reducing the step size by a factor of two and back stepping and so on. Each method of step-size variation is more adaptable to some types of functions rather than others. All of them basically try to find the point closest to the optimum before changing direction.

A Stochastic Minimization Technique

The final gradient method to be considered is a stochastic minimization technique suggested by Bertram (*4*). The discussion of this technique has been separated from the previous discussion of gradient methods because the partial derivatives and the step size used by this method are not arbitrarily determined; they are an integral part of the method.

The working formula for this technique is given by

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) - \frac{a_k}{c_k}\mathbf{y}(t_k)$$

The vector $\mathbf{y}(t_k)$ (a vector proportional to the partial derivatives) is generated by performing the following $n+1$ experiments:

$$\begin{bmatrix} x_1(t_k) \\ x_2(t_k) \\ \vdots \\ x_n(t_k) \end{bmatrix} \to y(t_k), \quad \begin{bmatrix} x_1(t_k) + c_k \\ x_2(t_k) \\ \vdots \\ x_n(t_k) \end{bmatrix} \to y_1(t_k), \ldots\ldots, \quad \begin{bmatrix} x_1(t_k) \\ x_2(t_k) \\ \vdots \\ x_n(t_k) + c_k \end{bmatrix} \to y_n(t_k)$$

and then forming

$$\mathbf{y}(t_k) = \begin{bmatrix} y_1(t_k) - y(t_k) \\ y_2(t_k) - y(t_k) \\ \vdots \\ y_n(t_k) - y(t_k) \end{bmatrix}$$

The scalars a_k and c_k which determine the magnitude of the step size are not entirely arbitrary, since they must meet the following conditions in order to guarantee convergence in the presence of process noise:

$$\sum_{k=1}^{\infty} a_k = \infty$$

$$\lim_{k\to\infty} c_k = 0$$

$$\sum_{k=1}^{\infty} a_k c_k < \infty$$

$$\sum_{k=1}^{\infty} \left(\frac{a_k}{c_k}\right)^2 < \infty$$

One elementary set of a_k's and c_k's which meets the required criteria is given by

$$a_k = A/k \qquad k = 1, 2, 3, \ldots.$$

$$c_k = B(k)^{\beta-1} \qquad 1 > \beta > \frac{1}{2}$$

A disadvantage of this method is that it requires $n + 1$ experiments for each point being explored. However the main disadvantage of this method as initially presented is that convergence may at times be extremely slow. Consequently a slight modification was introduced which not only greatly increased the rate of convergence but also greatly reduced the number of experiments required to obtain an optimal answer. The modification was merely that of holding k constant as one stepped from point to point in the state space rather than increasing k each time after obtaining a given point; the only time k was increased was when no improvement in the response function could be obtained with the k value.

Summary of Previous Discussions

The gradient (steepest descent or ascent) methods involve two distinct procedures; first, finding the best direction to change the independent variables, and second, moving a certain distance or step size in that direction. The choice of technique for each of these procedures is the desire for minimizing the time required for the optimization of any particular function and should be decided upon after examination of the nature of the function.

Numerical Example I.—The gradient techniques that have been presented were programed for an IBM-704 in order to compare efficiencies in the optimization of a complex nonlinear system involving a chemical reaction in a packed bed. Unfortunately the proprietory nature of this system is such that numerical details cannot be presented here. However since application of the search programs to the following simple three variable nonlinear function

$$y^* = x_1 + x_2 + x_3 + \frac{4}{x_1} + \frac{9}{x_2} + \frac{16}{x_3} - 18; \; x_1, x_2, x_3 > 0$$

gave results which were similar to those obtained for the simulated physical system, the calculated results for this system are presented below in a form suitable for comparing the efficiency of the various techniques. The criterion used for the comparison is that of minimizing the time required for the optimization. For the fixed-bed reactor it was found that the time required to carry out an experiment was much longer than the time required for all the remainder of the optimization technique. Therefore minimizing the number of experiments required to find the optimum was essentially the same as minimizing the time required. For simplicity this criterion of minimizing the number of experiments will also be used for comparing the techniques in the optimization of the simple function.

For this case it was found that the steepest descent techniques with the least-squares method for the estimation of the partial derivatives were most efficient for a rough optimization; that is the system was brought from any nonoptimal state to the neighborhood of an optimal state with a minimum number of experiments. Unfortunately once the vicinity of the optimal state was reached, an excessive number of experiments were required to achieve the final desired condition. In contrast the steepest descent procedure which used the factorial-design method for estimating the partial derivatives tended to take a great deal of time to move the system to the neighborhood of an optimal state, but once in the neighborhood the method converged rapidly to an optimal result. Hence it is not surprising that the best optimization technique turned out to be a least-squares method combined with the factorial-design method; the least-squares procedure brought the system to the vicinity of an optimal state, and the factorial-design procedure then brought the system quickly to a final optimal point.

A comparison of some of the numerical results is given in Table 1. The Ordinary Steepest Descent method (OSD), the Modified Steepest Descent method (MSD), and the Weighted Normalized Steepest Descent method (WNSD) all used least squares for estimating the partial derivatives. The OSD method used a gradient which was equal to the partial derivative vector and a step size which was always less than or equal to one. The MSD and WNSD methods used the weighted-normalized form for the gradient. The MSD method used an initial step size equal to one and then used the angle between a successive evaluations of the gradient vector to adjust the step size. The WNSD method also used an initial step size of one but then either doubled or divided the step size by four depending on the value of the newly obtained response function. The other methods are self-explanatory.

It can be seen that the better least-squares-steepest descent methods needed only twenty-one experiments to move the system from a highly nonoptimal state with a function value of 174.4531 to the vicinity of the optimal state (function

Table 1. Numerical Results for Various Minimization Techniques

Starting function value, $y = 174.4531$
Final desired function value, $y = 0.0$

	Total number of experiments	Function value
Combined wt.—normalized mod. factorial—design steepest descent	21	0.1875
	32	0.0010
	44	0.0001
Modified factorial—design steepest descent	10	171.4674
	15	164.0041
	131	0.5515
	176	0.0000
Weighted normalized steepest descent	6	169.2696
	10	65.9395
	17	1.7882
	21	0.1875
	555	0.0010
Modified steepest descent	6	170.1335
	10	138.5739
	15	34.3061
	21	0.4257
Ordinary steepest descent	6	168.4770
	24	114.3561
	55	24.6448
	67	0.1280
Stochastic minimization	66	0.5531
	100	0.0851
	9,044	0.0140
	57,940	0.0053

value of 0). However the convergence thereafter was very slow, and in the case of the stochastic minimization technique it was extremely slow. Even after completion of 57,940 experiments convergence was not obtained. The factorial-design procedure was much slower in the initial approach to the optimum, requiring 141 experiments, but only requiring thirty-five more experiments to reach the optimal result. In contrast the combined weighted-normalized-factorial design procedure required only a total of forty-four experiments to do the same optimization. Hence this combined approach is the most efficient of all of the techniques described.

TIME OPTIMAL CONTROL

Of increased importance as chemical processes become faster (or contact times shorter) is the need for search programs of the type discussed to locate a desired optimum condition plus some means of manipulating the control variables so that the search takes place as rapidly as possible and that the process moves as quickly as possible to this optimum. In particular this section treats the case where the immediate desired operating point of the process has been determined either by an off-line computation or as a step in a search program; it is then desired to force the process as close to this point as possible in the specific sense that a quadratic form containing the difference between the desired and actual operating points is minimized by a proper adjustment of the control variables. This minimization is to be carried out as quickly as possible, subject to the constraints of the process dynamics. In order to concentrate on the principles of the methods used it will be assumed that there are no constraints on the incremental amplitudes of the control variables and that the control variables can only change these amplitudes at periodic instants in time called *sampling instants.*

It has been shown (*1, 9*) that the dynamic behavior of many chemical operations in the immediate vicinity of a particular equilibrium condition can be represented by a set of ordinary linear differential equations with constant coefficients. In matrix form this set of equations can be written as

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\,\mathbf{x} + \mathbf{D}\,\mathbf{m} \qquad (1)$$

where **x** and **m** are assumed functions of time. The solution to Equation (1) follows in a straightforward manner to yield

$$\mathbf{x}(t) = \exp\,[\mathbf{A}t]\,\mathbf{x}(0) + \int_0^t \exp\,[\mathbf{A}(t-\lambda)]\,\mathbf{D}\,\mathbf{m}\,d\lambda \quad (2)$$

where

$$\exp\,[\mathbf{A}t] = \sum_{i=0}^{\infty} \frac{(\mathbf{A}t)^i}{i!}$$

$$\mathbf{x}(0) = \mathbf{x}(t) \text{ at } t = 0$$

If it is assumed that the inputs can be changed only at discrete time intervals $t = 0, \tau, 2\tau, ---, k\tau, ----$, Equation (2) may be simplified over any of the sampling periods to the form

$$\mathbf{x}(k+1) = \Phi\,\mathbf{x}(k) + \Delta\,\mathbf{m}(k) \quad (3)$$
$$k = 0, 1, 2, ----$$

where

$$\Phi = \exp\,(\mathbf{A}\,\tau)$$

$$\Delta = \{\textstyle\int_0^\tau \exp\,(\mathbf{A}\,\lambda)\,d\lambda\}\,\mathbf{D}$$

The terminology in Equation (3) is that $\mathbf{x}(k)$ indicates $\mathbf{x}(t)$ at the kth sampling instant and $\mathbf{x}(k+1)$ indicates $\mathbf{x}(t)$ one sampling period later. Since k is arbitrary, Equation (3) is a simple recurrence or difference equation for the system given by Equation (1). The elements of Φ and Δ are constants independent of k since **A** and **D** have only constant elements.

The time-optimal problem can be formulated in the following way:

Given a process governed by Equation (3) find that sequence of control vectors **m** (k) $(k = 0, 1, 2, ---)$ such that the following criterion is minimized as $N \to \infty$:

$$J = \sum_{1}^{N} [\mathbf{x}^d - \mathbf{x}(k)]^T\,\mathbf{Q}\,[\mathbf{x}^d - \mathbf{x}(k)] \quad (4)$$

The terminology of Equation (4) requires some elucidation before the meaning of minimizing J can be discussed. J is a quadratic form in which $\mathbf{x}^d$ is the set of desired dependent variable conditions which the process is to achieve. The expansion of Equation (4) merely leads to a weighted sum of squares sequence with the weighting determined by the elements of **Q** (with the assumption that **Q** is a diagonal matrix). If **Q** is the identity matrix **I**, the weighting factors are unity. Thus minimizing J tends to minimize the sum of squares between $\mathbf{x}^d$ and **x**. As an illustration consider the case of $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. Assuming that $x_2 = 5x_1$ always and it is desired that $x_1 = 2\%$ and $x_2 = 10\%$, one can write a quadratic error formula as

$$J = (2 - x_1)^2 + (10 - 5x_1)^2$$

The control would proceed in a manner to minimize this J. In the matrix notation of Equation (4) this could also be written as

$$J = \left\{\begin{bmatrix} 2 \\ 10 \end{bmatrix} - \begin{bmatrix} x_1 \\ 5x_1 \end{bmatrix}\right\}^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \left\{\begin{bmatrix} 2 \\ 10 \end{bmatrix} - \begin{bmatrix} x_1 \\ 5x_1 \end{bmatrix}\right\}$$

As a final point it should be noted that the minimization of Equation (4) implicitly involves the difference Equation (3) of the dynamic system since $\mathbf{x}(k)$ is involved.

The desired result of the minimization of Equation (4) would be to yield a sequence of control vectors

$$\mathbf{m}(k), \mathbf{m}(k+1), \mathbf{m}(k+2), --- \quad (5)$$

which will force the dynamic system to move from the starting or initial condition $\mathbf{x}(t_0)$ to the closest possible point near $\mathbf{x}^d$. Unfortunately there are many possible linear transformations,

Table 2. Variation in Control Values and Response as Function of Sampling Time

Sampling times	Control values x_0	Control values $(y_\tau - \beta)/\alpha$	Response x_0
0τ	0	0	0.25464
1τ	0.21828	0.76370	0.44376
2τ	0.08949	0.49771	0.41898
3τ	0.06414	0.46560	0.40583
4τ	0.04589	0.44826	0.39787
5τ	0.03242	0.43771	0.39272
6τ	0.02267	0.43090	0.38930
7τ	0.01574	0.42637	0.38699
8τ	0.01089	0.42330	0.38541
9τ	0.00751	0.42122	0.38433
10τ	0.00518	0.41979	0.38359
20τ	0.00012	0.41674	0.38201
$\infty\tau$	0.00000	0.41666	0.38197

depending on the choice of $\mathbf{m}(k)$, that will perform this operation. Thus the minimization is really an infinite stage-decision procedure with the choice of $\mathbf{m}(k)$ at the kth stage depending both on the preceding and succeeding decisions. The control objective is to find that sequence, among the possible sets, which will be optimal in the sense that the time of operation of the system is minimized. This particular sequence will be denoted as

$$\mathbf{m}^{\circ}(k), \mathbf{m}^{\circ}(k+1), \mathbf{m}^{\circ}(k+2), ---- \qquad (6)$$

The determination of the sequence (6) can be obtained by a technique described by Kalman, Lapidus, and Shapiro (9) termed "dynamic programing." Dynamic programing was developed by Bellman (3) and represents a major contribution to the handling of multistage decision processes. In effect it replaces the multistage problem with a series of single-stage problems. To illustrate the mechanics of this process it is simpler to take J equal to

$$J = \sum_{1}^{N} \mathbf{x}^T(k)\, \mathbf{Q}\, \mathbf{x}(k) \qquad N \to \infty$$

rather than Equation (4). In effect this equation is merely a shifted or transformed version of (4) with $\mathbf{x}^d$ removed. To minimize J, N partial derivatives of J are taken with respect to $\mathbf{m}(k)$ $(k = 0, 1, 2, ---, N-1)$ and the results set equal to zero starting with $k = N-1$. Note that $\mathbf{m}(k)$ appears in the equation for J in terms of the difference Equation (3) and that the entire procedure of differentiation starts at the final end of the control sequence and progresses backwards to the initial end $(k = N-1, N-2, ----, 2, 1, 0)$. Performing the differentiation with respect to $\mathbf{m}$ $(N-1)$ one obtains

$$\frac{\partial J}{\partial \mathbf{m}(N-1)} = 2\frac{\partial \mathbf{x}^T(N)}{\partial \mathbf{m}(N-1)}\mathbf{Q}\,\mathbf{x}(N) = 0 \qquad (7)$$

But since a control signal at the kth sampling instant cannot affect the behavior of the dependent variables before this time $t < k\tau$, it follows in Equation (7) that

$$\frac{\partial \mathbf{x}^T(k)}{\partial \mathbf{m}(N-1)} = 0 \qquad k < N$$

and

$$\frac{\partial \mathbf{x}^T(k)}{\partial \mathbf{m}(N-1)} = \Delta^T \qquad k = N$$

where Equation (3) has been used. Equation (7) can thus be written as

$$\Delta^T \mathbf{Q}\, \mathbf{x}(N) = 0$$

and using Equation (3) one gets

$$\Delta^T \mathbf{Q}\, [\Phi\, \mathbf{x}(N-1) + \Delta\, \mathbf{m}(N-1)] = 0$$

Solving for $\mathbf{m}(N-1)$ one gets

$$\mathbf{m}(N-1) = -\,[(\Delta^T \mathbf{Q}\, \Delta)^{-1} \Delta^T \mathbf{Q}\, \Phi]\, \mathbf{x}(N-1)$$

which may be written in shortened form as

$$\mathbf{m}(N-1) = \mathbf{C}^{(N-1)}\, \mathbf{x}(N-1) \qquad (8)$$

Since this is the solution for $\mathbf{m}(N-1)$, it may be substituted into Equation (3) to yield

$$\mathbf{x}(N) = [\Phi + \Delta\, \mathbf{C}^{(N-1)}]\, \mathbf{x}(N-1)$$

or

$$\mathbf{x}(N) = \Psi^{(N-1)}\, \mathbf{x}(N-1) \qquad (9)$$

The process is now repeated with J differentiated with respect to $\mathbf{m}(N-2)$. As above

$$\frac{\partial J}{\partial \mathbf{m}(N-2)} = 0 = 2\frac{\partial \mathbf{x}^T(N)}{\partial \mathbf{m}(N-2)}\mathbf{Q}\,\mathbf{x}(N) + 2\frac{\partial \mathbf{x}^T(N-1)}{\partial \mathbf{m}(N-2)}\mathbf{Q}\,\mathbf{x}(N-1)$$

Substituting Equation (9) and rearranging one obtains

$$\frac{\partial \mathbf{x}^T(N-1)}{\partial \mathbf{m}(N-2)}[\Psi^{(N-1)T}\, \mathbf{Q}\, \Psi^{(N-1)} + \mathbf{Q}]\, \mathbf{x}(N-1) = 0$$

and when one lets

$$\mathbf{P}^{(1)} = [\Psi^{(N-1)T}\, \mathbf{Q}\, \Psi^{(N-1)} + \mathbf{Q}]$$

there results

$$\frac{\partial \mathbf{x}^T(N-1)}{\partial \mathbf{m}(N-2)}\mathbf{P}^{(1)}\, \mathbf{x}(N-1) = 0 \qquad (10)$$

But this equation is identical with (7) except that $\mathbf{P}^{(1)}$ replaces $\mathbf{Q}$ in (7) and that the index on k has been dropped by 1. As a result it immediately follows in analogy with (8) and (9) that

$$\mathbf{m}(N-2) = \mathbf{C}^{(N-2)}\, \mathbf{x}(N-2) \qquad (11)$$

$$\mathbf{x}(N-1) = \Psi^{(N-2)}\, \mathbf{x}(N-2) \qquad (12)$$

This entire process can be continued by differentiating J with respect to $\mathbf{m}(N-3)$, $\mathbf{m}(N-4)$, -----, but it soon becomes evident that an iteration pattern is formed, given by

$$\mathbf{P}^{(i)} = \Psi^{(N-i)T}\, \mathbf{P}^{(i-1)}\, \Psi^{(N-i)} + \mathbf{Q}$$

$$\mathbf{C}^{[N-(i+1)]} = -(\Delta^T \mathbf{P}^{(i)}\, \Delta)^{-1}\, \Delta^T \mathbf{P}^{(i)}\, \Phi \qquad (13)$$

$$i = 0, 1, 2, ---$$

$$\Psi^{[N-(i+1)]} = \Phi + \Delta\, \mathbf{C}^{[N-(i+1)]}$$

with $\mathbf{P}^{(0)} = \mathbf{Q}$ and $\Psi^{(N)} = 0$.

The iteration pattern of Equation (13) can be used to successively calculate the sequence of control vectors given by

$$\mathbf{m}(N-i) = \mathbf{C}^{(N-i)}\, \mathbf{x}(N-i) \qquad (14)$$

$$i = 1, 2, ---$$

in a reverse manner. In other words the last signal $\mathbf{m}(N-1)$ as applied to $\mathbf{x}(N-1)$ is calculated first and then $\mathbf{m}(N-1)$, $\mathbf{m}(N-3)$, ---. It can be shown that as long as $\mathbf{Q}$ is positive definite and $(\Delta^T \mathbf{Q}\, \Delta)^{-1}$ exists, then the matrices $\mathbf{C}^{(N-i)}$ always converge to a constant matrix $\mathbf{C}$ as $N \to \infty$. Thus for N large enough it is possible to write

$$\mathbf{m}(k) = \mathbf{C}\, \mathbf{x}(k)$$

or, since $\mathbf{m}(k)$ are optimal in the sense that J has been minimized

$$\mathbf{m}^{\circ}(k) = \mathbf{C}\, \mathbf{x}(k) \qquad (15)$$

Note that this is an equation which allows the calculation of the successive control vectors $\mathbf{m}^{\circ}(k)$ from the $\mathbf{x}(k)$ obtained from Equation (3) in the form

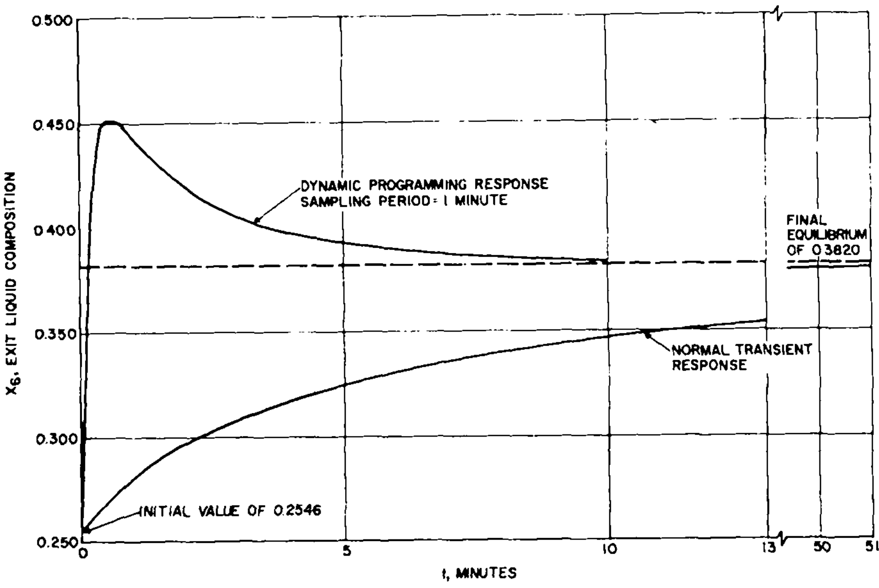

Fig. 1. Control and normal response of exit liquid composition from plate type of absorber.

$$\mathbf{x}(k) = \Phi\,\mathbf{x}(k-1) + \Delta\,\mathbf{m}^{\circ}(k-1) \qquad (16)$$

When Equation (4) is used to define J, an analogous procedure leads to

$$\mathbf{m}^{\circ}(k) = \mathbf{C}\,\mathbf{x}(k) + \mathbf{B}\,\mathbf{x}^{d} \qquad (17)$$

where $\mathbf{C}$ is as given in (15) and

$$\mathbf{B} = (\rho^{T}_{\cdot\cdot}\,\mathbf{Q}\,\rho_{\cdot\cdot})^{-1}\rho_{\cdot\cdot}\,\mathbf{Q}$$
$$\rho_{\cdot\cdot} = (\mathbf{I} - \Psi)^{-1}\Delta$$

In this last equation Ψ is obtained as a part of the iteration pattern of (13).

The final result is now available in the form of the dynamic difference Equation (3) and the optimal set of control vectors Equation (17). Starting with $\mathbf{x}(0)$ as an initial condition on the dynamic system one can use (17) to calculate the initial control vector $\mathbf{m}^{\circ}(0)$. Using (3) one can calculate the value of $\mathbf{x}(1)$ (corresponding to the state of the system at the end of one sampling period) and then with (17), $\mathbf{m}^{\circ}(1)$. This is continued until $\mathbf{x}(k)$ approaches or becomes equal to $\mathbf{x}^{d}$.

Equation (17) shows that the optimal control signal $\mathbf{m}^{\circ}(k)$ is a linear function of the state of the system $\mathbf{x}(k)$. This was to be expected in view of the quadratic performance index and the linear dynamics considered.

An important point resulting from this development is that the performance index, Equation (4), is minimized, and the sequence of optimal control variables is calculated by observing (or measuring) the system at the times

$$t_0,\ t_0 + \tau,\ t_0 + 2\tau,\ - - - - -$$

Equation (15) shows that the feedback principle is automatically included in the optimization procedure. Actually one might suggest that the use of Equations (3) and (15) is sufficient to predict the entire future evolution of the process. However because of a possibly inaccurate knowledge of the transition matrix, unknown disturbances acting on the process, and possible random effects of various kinds the prediction based on (3) will become less and less correct as the prediction interval increases. By remeasuring the state of the system at the sampling instants, on the assumption that τ has been chosen small enough so that the one step prediction based on (3) is sufficiently accurate, the prediction errors are corrected and the control variables assume very nearly their optimal values at all times. This mode of operation defines a feedback control system.

It is worthwhile at this point to compare the present approach with the more standard methods of designing sampled-data control systems. At this time the standard approach to the design of a system of the type described is based on the z transform (11). One of the drawbacks of this method is that it does not provide a direct way of investigating system behavior at instants of time other than the sampling instants. The method discussed here is not subject to this difficulty (5). In addition the numerical design calculations are carried out by means of the technique of dynamic programing (3) which is ideally suited for digital computation. Solutions can be obtained rapidly by iterating simple expressions of the form of Equation (13). This is a most important consideration in the design of automatically optimized systems, since the parameters characterizing the control action must be recomputed to maintain optimal performance if the system is one in which the dynamic characteristics change. It is also to be noted that the design has been accomplished without use of the spectral factorization technique to solve a Wiener-Hopf equation required by conventional control system design methods (12).

Numerical Example II.—To illustrate the principles discussed above a numerical example is herein presented as applied to the transient behavior of a plate type of adsorption tower. The necessary transient equations were presented by Lapidus and Amundson (1) and will not be rederived.

A material balance around the nth plate leads to

$$L\,[x_{n-1} - x_n] + G\,[y_{n+1} - y_n] = H\frac{dy_n}{dt} + h\frac{dx_n}{dt} \qquad (18)$$

It is implicit in this balance that $x_n = x_n(t)$ and $y_n = y_n\,(t)$ with t = time. Making use of a linear equilibrium relationship one gets

$$y_n = \alpha\, x_n + \beta \qquad (19)$$

Equation (18) becomes

$$\frac{dx_n}{dt} = \frac{d}{e}\,x_{n-1} - \left(\frac{d+1}{e}\right)x_n + \frac{1}{e}\,x_{n+1}$$

$$n = 1, 2, - - -, N \qquad (20)$$

where $d = LG/\alpha$ and $e = \dfrac{H\alpha + h}{G\alpha}$. To this material balance must be added the initial condition

$$x_n(0) = A_n,\ t = 0 \qquad (21)$$

and the feed conditions

$$x_n(t) = x_0(t),\ n = 0$$
$$y_n(t) = y_{N+1}(t),\ n = N + 1 \qquad (22)$$

Equations (20), (21), and (22) represent a set of linear dynamic equations and can be written in matrix form as

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x} + \mathbf{D}\mathbf{m} \qquad (23)$$

where

$$\mathbf{A} = \begin{bmatrix} -\left(\dfrac{d+1}{e}\right) & 1/e & & \\ d/e & -\dfrac{(d+1)}{e} & 1/e & O \\ & d/e & & \\ O & & & 1/e \\ & & d/e & -\dfrac{(d+1)}{e} \end{bmatrix} = \text{tridiagonal matrix}$$

$$\mathbf{D} = \begin{Bmatrix} d/e & 0 \\ 0 & , \\ , & , \\ , & , \\ , & \\ , & 0 \\ 0 & 1/e \end{Bmatrix},\quad \mathbf{x} = \begin{Bmatrix} x_1 \\ x_2 \\ , \\ , \\ , \\ , \\ x_n \end{Bmatrix},\quad \mathbf{m} = \begin{bmatrix} x_0 \\ \dfrac{y_{N+1} - \beta}{\alpha} \end{bmatrix}$$

To illustrate the dynamic programing optimization technique the following numerical parameters were chosen:

$$N = 6.0 \text{ plates}$$
$$\alpha = 0.72$$
$$\beta = 0$$
$$L = 40.8 \text{ lb./min.}$$
$$G = 66.7 \text{ lb./min.}$$
$$H = 1.0$$
$$h = 75$$

These lead to

$$d = 0.85000$$
$$1/e = 0.634115$$

The dynamic problem of interest corresponds to an initial equilibrium distribution in the tower resulting from feed inputs of $x_0 = 0$ (pure liquid) and $y_{N+1} = 0.2$ lb. solute/lb. inert and then following the time response when the gas composision is stepped to $y_{N+1} = 0.3$ lb. solute/lb. inert at $t = 0$. Corresponding to the initial feed compositions Equation (20) can be solved with $dx_n/dt = 0$ to yield the corresponding equilibrium compositions on each plate. These are

$$\begin{array}{ll} x_1 = 0.0613266 & \\ x_2 = 0.1134541 & \\ x_3 = 0.1577626 & \\ x_4 = 0.1954247 & \text{for } t = 0 \\ x_5 = 0.2274376 & x_0 = 0 \qquad (24) \\ x_6 = 0.2546485 & y_{N+1} = 0.2 \end{array}$$

as initial conditions $\mathbf{x}(t = 0)$. In the same way but with the final stepped gas stream input

$$
\begin{aligned}
x_1 &= 0.0919898 \\
x_2 &= 0.1701812 \\
x_3 &= 0.2366438 \\
x_4 &= 0.2931371 && \text{for } t \to \infty \\
x_5 &= 0.3411564 && x_0 = 0 \qquad (25) \\
x_6 &= 0.3819727 && y_{N+1} = 0.3
\end{aligned}
$$

These compositions correspond to the final desired condition or $\mathbf{x}^d$.

The question now arises as to whether the dynamic programing optimization can be used to force the system to move from the initial state (24) to the final state (25) in a manner faster than the normal response of the system. The normal response can be obtained by numerically integrating (20) with (24) as initial values and feeds corresponding to $x_0 = 0$, $y_{N+1} = 0.3$. This calculation was performed on the IBM-704 with a Runge-Kutta-Gill integration procedure. The result is shown in Figure 1 in terms of x_6 the fat liquid composition. It is noticed that even after 50 min. the exit composition has not attained the desired value, that is x_6 (50 min.) = 0.37998801 vs. x_6 $(t = \infty)$ = 0.3819727.

To illustrate the dynamic programing the differential equations of (23) are first converted to the basic difference form

$$\mathbf{x}(k+1) = \Phi\, \mathbf{x}(k) + \Delta\, \mathbf{m}(k) \qquad (26)$$

where, using the numbers given and a sampling period τ = 1 min., one obtains

$$\Phi = \sum_{i=0}^{20} \frac{(\mathbf{A}\tau)^i}{i!} = \begin{bmatrix} 0.36537 & 0.21952 & 0.06770 & 0.01407 & 0.00220 & 0.00027 \\ 0.18659 & 0.42292 & 0.23148 & 0.06958 & 0.01430 & 0.00220 \\ 0.04891 & 0.19676 & 0.42452 & 0.23168 & 0.06958 & 0.01407 \\ 0.00864 & 0.05027 & 0.19693 & 0.42452 & 0.23148 & 0.06770 \\ 0.00115 & 0.00878 & 0.05027 & 0.19676 & 0.42292 & 0.21952 \\ 0.00012 & 0.00115 & 0.00864 & 0.04891 & 0.18659 & 0.36537 \end{bmatrix}$$

$$\Delta = \{\textstyle\int_0^\tau \phi(\lambda)\, d\lambda\}\, \mathbf{D} = \begin{bmatrix} 0.33080 & 0.00003 \\ 0.07255 & 0.00033 \\ 0.01165 & 0.00280 \\ 0.00146 & 0.01897 \\ 0.00015 & 0.10042 \\ 0.00001 & 0.38917 \end{bmatrix}$$

In (26) $\mathbf{m}(k)$ is a 2×1 vector containing two control variables. These variables are x_0 and y_{N+1}. Given a value of $\mathbf{x}(k)$ and $\mathbf{m}(k)$ this equation can be used to calculate the liquid compositions $\mathbf{x}(k+1)$ at the end of one sampling period. In addition the equation

$$\mathbf{m}^o(k) = \mathbf{C}\, \mathbf{x}(k) + \mathbf{B}\, \mathbf{x}^d \qquad (27)$$

can be used to calculate the control vector $\mathbf{m}^o(k)$ given $\mathbf{x}(k)$ and $\mathbf{x}^d$ and the 2×6 matrices **C** and **B**. These latter matrices can be obtained from the dynamic programming iteration process previously described. For τ = 1 min. the sequence converges in sixteen iterations to yield the values, correct to six decimal places

$$\mathbf{C} = \begin{bmatrix} -1.22763 & -1.13746 & -0.69774 & -0.37809 & -0.16869 & -0.03888 \\ -0.03004 & -0.14597 & -0.33898 & -0.61705 & -0.98475 & -1.09026 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 2.18511 & 1.49812 & 0.91418 & 0.41783 & 0.00406 & 0.36268 \\ -0.37349 & 0.14998 & 0.59493 & 0.97315 & 1.29463 & 1.5678 \end{bmatrix}$$

With this information in hand (26) and (27) are used successively to move the absorption tower in an optimal dynamic manner. Starting with $\mathbf{x}(0)$ as given in (24) and $\mathbf{x}^d$ as given in (25), (27) is used to calculate the control values $\mathbf{m}^o(0)$. This $\mathbf{m}^o(0)$ and $\mathbf{x}(0)$ are used in (26) to calculate $\mathbf{x}(1)$. Equation (27) is next used to calculate $\mathbf{m}^o(1)$ and then (26) for $\mathbf{x}(2)$. This is continued as long as desired. On the IBM-704 the entire sequence of calculations described here required about 30 sec. The results are shown in Figure 1. At the end of only 10 min. the exit liquid composition is already x_6(10 min.) = 0.3835991. This contracts very markedly with the normal transient response. Table 2 presents some of the numerical data obtained.

It should be pointed out that from a physical point of view to control by varying the inlet compositions is not very useful, instead one would normally maintain the compositions fixed and vary the flow rates. Such a procedure in turn leads to certain theoretical difficulties, since the **A** matrix in (23) becomes time varying. In subsequent publications from this laboratory various means of handling this situation·will be discussed.

SUMMARY AND CONCLUSIONS

An important feature of process optimization is to bring the process from an undesired state to a desired state in a minimum amount of time. In other words one tries to maximize profits (or minimize cost) by staying away from a nonoptimum condition for as short a time as possible. The present paper shows that is is possible to carry out a search of the possible settings of the independent variables of the process in such a way as to quickly locate an extremal of the possible values of a chosen objective function. A changing (or variable) search algorithm may be used for this purpose, wherein speed is emphasized at the beginning of the search and accuracy at the end of the search. A computational technique carried out on a real-time basis that is founded on the concept of dynamic programing is also described, whereby it is possible to perform the individual search steps in a minimum amount of time. In this way fast execution of the search programs is assured together with rapid location and tracking of the desired extremal.

NOTATION

A	=	n x n square matrix
A, B	=	positive constants usually set equal to unity
D	=	n x m matrix
G	=	gradient, vector proportional to partial derivatives of y^* with respect to x_i
G	=	inert gas rate
H	=	vapor holdup on any plate
h	=	liquid holdup on any plate
L	=	absorbent rate
m	=	m x 1 vector containing m inputs or control variables of process
N	=	number of plates
Q	=	symmetrical positive definite matrix
t_k	=	time of kth experiment
x	=	independent variable vector
$\mathbf{x}(k)$	=	set of dependent variables at kth sampling instant
x_n	=	composition of liquid leaving nth plate
y_n	=	composition of vapor leaving nth plate
$y(t_k)$	=	linear estimate of response function
$y^*(t_k)$	=	measured value of function
λ	=	positive scalar representing magnitude of step size
τ	=	constant sampling period
Φ	=	transition matrix

Superscript

T	=	transpose of vector

LITERATURE CITED

1. Amundson, N. R., and Leon Lapidus, *Ind. Eng. Chem.*, **42**, 1071 (1950).
2. Beckenbach, E. F., "Modern Mathematics for the Engineer," McGraw-Hill, New York (1956).
3. Bellman, R. E., "Dynamic Programming," Princeton Univ. Press, Princeton, New Jersey (1957).
4. Bertram, J. E., *Trans. Am. Inst. Elec. Engrs.*, **79**, No. 2, p. 485 (1960).
5. ———, and R. E. Kalman, *J. Franklin Inst.*, **267**, 405 (May, 1959).
6. Box, G. E. P., and K. B. Wilson, *J. Royal Statistical Soc.*, **B13**, 1 (1951).
7. Dennis, J. B., "Mathematical Programming and Electrical Networks," J Wiley, New York (1959).
8. Feldbaum, A. A., *Automation and Remote Control*, **19**, No. 8, p. 718 (August, 1958).
9. Kalman, R. E., Leon Lapidus, and Eugene Shapiro, "Proceedings of the Joint Symposium on Instrumentation and Computation," London, England (May, 1959).
10. Marquardt, D. W., *Chem. Eng. Progr.*, **55**, 65 (1959).
11. Ragazzini, J. R., and G. F. Franklin, "Sampled-Data Control Systems," McGraw-Hill, New York (1958).
12. Truxal, J. G., "Control System Synthesis," McGraw-Hill, New York (1955).

Manuscript received April 4, 1960; revision received October 26, 1960; paper accepted October 27, 1960. Paper presented at A.I.Ch.E. Mexico City meeting.