A Lagrangian Algorithm for Equality Constrained Generalized Polynomial Optimization

GARY E. BLAU
Dow Chemical Co., Midland, Michigan
and
DOUGLASS J. WILDE
Stanford University, Stanford, California

A procedure for optimizing generalized polynomial functions occurring widely in engineering design problems is presented. Equality constraints of the same form can be handled. The technique involves driving to zero the gradient of a Lagrangian function by a Newton-Raphson method. A nonlinear transformation so simplifies the derivatives needed for Newton-Raphson iteration that they are given in closed form. The initial estimate needed to start the algorithm need not be feasible. Successive systematic adjustment of Lagrange multipliers is accomplished, and an unambiguous procedure for starting multipliers is given. All implicit computations are linear. Convergence is not proven, but successful computational results for the design of a reactor exchanger pump system in eight variables and with five constraints are presented.

This article gives a new procedure for finding optimal process designs of any system whose economic objective function and constraining physical equations are generalized polynomials. Although the term generalized polynomial may be unfamiliar to many engineers, the functions themselves should not, for they are generated by any phenomenon which plots as a straight line in log-log coordinates. Most design problems can often be fitted neatly into this theoretical framework, perhaps after straightforward changes of variable have removed fractions and radicals (*4*). Published examples of systems modeled by generalized polynomials already include electrical transformers (*4*), heat exchangers (*4, t*), and chemical reactors with external cooling (*6*). Process design texts (*7*) are full of further possibilities.

Present optimization techniques (*3*) are of little practical value for generalized polynomial problems because the functions are highly nonlinear and only rarely convex. The first attempts to optimize such functions led to the elegant theory of geometric programming (*4*), which suggests how to optimize posynomials, a subclass of generalized polynomials in which no negative terms are permitted. Geometric programming replaces the original problem by an auxiliary one involving optimizing a convex function subject only to linear constraints. The latter problem is sometimes easier to solve than the former, especially when the number of terms is not much greater than the number of variables. An efficient nonlinear programming algorithm for the auxiliary problem of geometric programming has already been applied to the design of vapor condensers (*5*). This approach fails, however, if any terms in the objective or constraint functions are negative.

The existing theory of polynomial optimization (*1, 3*) gives no way to compute optima except in the special case where there is exactly one more term than there are independent variables. Also the theory is formulated in terms of inequality constraints, although the physical restrictions occurring in practice are more often strict equalities. This article gives an optimization algorithm, suitable for equality constrained problems, which has exhibited rapid convergence on a realistic reactor-exchanger-pump system with nine variables (*6*).

The main idea is to use a Newton-Raphson procedure (*3*) to drive to zero the components of the gradient of a Lagrangian function formed from the logarithms of the original objective function and the constraints. A nonlinear transformation, which amounts to substituting a weighting variable for each term, makes the Lagrangian gradient linear in the weights as well as in the Lagrange multipliers, although bilinear in the two sets of variables taken together. The polynomial form of all the functions makes available, in closed form, the derivatives needed for the Newton-Raphson iteration. To begin, one selects a trial primal solution, not necessarily satisfying the constraints exactly. This immediately gives the initial values of all the weight variables, which, when substituted into the Lagrangian gradient, gives a set of functions linear in the unknown Lagrange multipliers. By straightforward and well-known matrix calculations, the initial values of the multipliers are chosen to minimize the sum of squares of these functions.

Substitution of these values of weights and Lagrange multipliers into simple formulas gives the numbers needed for a Newton-Raphson adjustment, obtained by solving a nonsingular set of linear equations. This gives new values of the Lagrange multipliers and of the original primal variables. When the latter are transformed into new weights, the Newton-Raphson procedure is repeated.

Eben and Ferron (*8*) have a different computation scheme which is also applicable to generalized polynomials. They use the method of successive substitutions (*9*) to solve a set of linear equations, similar to those in this article, although they are derived in a different way. The beauty of their approach is its simplicity and adaptability to problems involving functions not generalized polynomials, for example, those with exponential terms. The Lagrangian approach in the present article is quite different, using second-order conditions usually yielding more rapid convergence but requiring differentiability. Also the present method requires neither elimination of nonlinear equations nor selection of an independent system of nonlinear transformations.

Since exposition and justification of the algorithm requires only straightforward differentiation, transformation, and solution of linear equations, the paper is self-contained. Global convergence is, of course, ruled out because of the possibility of multiple zeros of the Lagrangian gradient. Because, as for all other methods for solving this nonconvex problem, no proof of convergence is available, computational results for two numerical examples are given. Since they indicate that the algorithm's high performance is not unduly sensitive to the initial choice of primal values, the algorithm merits at least a cautious try by design engineers.

CHEMICAL REACTOR DESIGN

Before examining the algorithm consider the problem of finding the best design of the chemical reactor system shown in Figure 1. In this problem, originally formulated by Westbrook and Aris (*10*), a feed stream q enters the reaction vessel of a fixed temperature T_I. It is required to produce a certain commodity E at a rate P by means of two parallel exothermic reactions:

$$A \xrightarrow{k_1} B$$

$$A + D \xrightarrow{k_2} E$$

The heat generated by the reactions is removed by an external heat exchanger while isothermal temperature control is maintained by circulating a bypass stream. The problem is to design the system to produce P while optimizing some economic criterion.

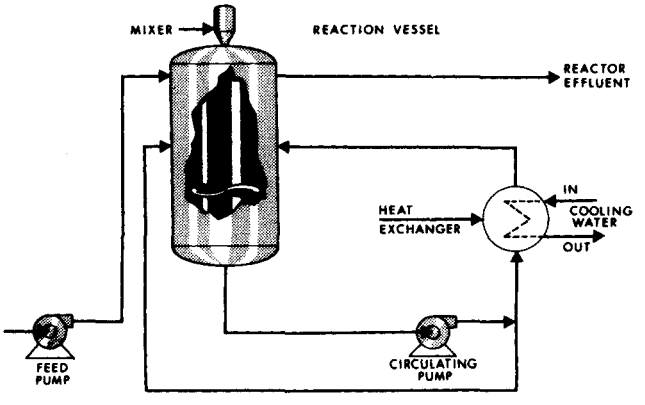

Fig. 1. Chemical reaction system.

In reference *6* this problem is formulated as a generalized polynomial program with nine variables and six inequality constraints. Briefly the methodology proceeds as follows: (a) The kinetic rate expressions are specified and used to write steady state material and energy balances. (b) Conventional design relations are employed to size the reactor vessel, exchanger, pumps, and all other pieces of equipment. (c) An economic criterion is specified and expressed in terms of the parameters appearing in (a) and (b). Some simplifying assumptions and algebraic manipulations are usually necessary so that the model consists exclusively of generalized polynomials. Anyone contemplating the case of generalized polynomial programming for plant design should become familiar with reference *6*.

A severe limitation on the procedure was the necessity to transform, by heuristic methods, the physically realistic equality constraints into inequalities suitable for analysis by the algorithm described in reference *11*. The technique described in this paper permits the use of equality constraints so that problem formulation is greatly simplified. Consider this optimal reactor design problem for profit maximization at $T = 200°F$. formulated as an equality constrained generalized polynomial program:

$$\text{minimize } -g_0 = 0.5(O_C - S) \quad \sigma = -1 \tag{1}$$

where

$$S = 492{,}000 + 556\, c_A\, d_V^3$$

$$O_C = 18{,}000 + 0.19I + 10.5 d_V^3 + 7.32 \times 10^{-10} \frac{w_o^{2.85}}{A^{1.70}} + 0.0189 w_i + 2.12 \times 10^{-10} \frac{w_i^{2.80}}{A^{1.80}} + 232q - 410 q c_A - 82000$$

$$I = 21.5 W^{0.782} + 5500 + 550 d_V + 2030 d_V^{0.9} + 647 q^{0.467} + 0.184 \frac{w_o^{1.33}}{A^{0.8}} + 1060 A^{0.546}$$

subject to the following:

1. Material balance constraint:

$$g_1 = \frac{129.4}{d_V^3} + \frac{105}{q} = 1 \tag{2}$$

2. Energy balance constraint:

$$g_2 = 1.03 \times 10^5 \frac{c_A d_V^3}{w_o \Delta T_o} + 1.20 \times 10^6 \frac{1}{w_o \Delta T_o} = 1 \tag{3}$$

3. Reactor vessel design constraints:

$$g_3 = 4.68 \frac{d_V^3}{W} + 6.13 \frac{d_V^2}{W} + 160.5 \frac{d_V^2}{W} = 1 \tag{4}$$

$$g_4 = 1.79 c_A + 3.02 \frac{d_V^3 c_A}{q} + \frac{35.7}{q} = 1 \tag{5}$$

4. Exchanger design constraint:

$$g_5 = 1.22 \times 10^{-3} \frac{w_o \Delta T}{w_i^{0.80} A^{0.20}} + 1.67 \times 10^{-3} \frac{w_o^{0.4} \Delta T}{A^{0.43}} + 3.6 \times 10^{-5} \frac{w_o \Delta T}{A} + 2.0 \times 10^{-3} \frac{w_o \Delta T}{w_i} + 4.0 \times 10^{-3} \Delta T = 1 \tag{6}$$

TABLE 1. NOMENCLATURE FOR REACTOR DESIGN PROBLEM

A	area of heat transfer, sq. ft.
c_A	concentration of component A, lb.-moles/cu. ft.
d_V	diameter of exchanger tubes, ft.
I	fixed plant investment, $
k_1, k_2	reaction rate constants, hr.$^{-1}$
O	operating cost, $/yr.
P^c	production rate of D, lb.-moles/yr.
q	reactor volumetric flow rate, cu.ft./hr.
T	temperature of process stream, °F.
w_i, w_o	mass flow rate inside (outside) tubes, lb./hr.
W	weight of reactor steel, lb.
ΔT	temperature driving force, °F.-
π	annual profit, $/yr.

where the parameters are defined in Table 1. The complete formulation of this problem from first principles is contained in reference 6. The computational aspects of the solution will be presented after the algorithm has been examined and illustrated with a simple example.

DESCRIPTION OF METHOD

This section defines the polynomial optimization problem and describes the algorithm for solving it. Justification of the procedure is presented later, following a sample calculation and discussion of computational experience.

Consider $\mathbf{M}+1$ generalized polynomials $\mathbf{g}_m$ defined by

$$\mathbf{g}_m \equiv \sum_{t=1}^{T_m} \sigma_{mt} c_{mt} \prod_{n=1}^{N} x_n^{a_{mtn}} \qquad m = 0, 1, \ldots, M \tag{7}$$

The strictly positive, finite independent variables $\mathbf{x}_n$ are to be chosen

$$0 < x_n < \infty \qquad n = 1, \ldots, N \tag{8}$$

whereas the positive real constants

$$c_{mt} > 0 \qquad t = 1, \ldots, T_m \tag{9}$$

the arbitrary real exponents $\mathbf{a}_{mtn}$, and the signum functions

$$\sigma_{mt} = \pm 1 \tag{10}$$

are presumed known in advance. The equality constrained generalized polynomial minimization problem is

$$\min_{\mathbf{x}} g_0 \tag{11}$$

subject to

$$g_m = 1 \qquad (m \neq 0) \tag{12}$$

To initiate the algorithm, choose finite, positive values $\mathbf{x}_n^\circ$ of the $\mathbf{x}_n$ not necessarily satisfying the constraint Equations (12), and use Equation (7) to compute corresponding values $\mathbf{g}_m^\circ$ of the polynomials. Then calculate a weight $\mathbf{w}_{mt}^\circ$, one for each term, by

$$w_{mt}^\circ = \begin{cases} (\sigma g_0^\circ)^{-1} c_{ot} \prod_{n=1}^{N} (x_n^\circ)^{a_{otn}} & \text{for } m = 0 \\ (g_m^\circ)^{-1} c_{mt} \prod_{n=1}^{N} (x_n^\circ)^{a_{mtn}} & \text{for } m = 1, \ldots, M \end{cases} \tag{13}$$

where the sign of g_0 at the desired minimum $\mathbf{x}^*$

$$\sigma \equiv \text{sign}\,(g_0^*) \tag{14}$$

is assumed known in advance and $\sigma g_0^\circ > 0$. Next evaluate the following sums:

$$s_{nm}^\circ \equiv \sum_{t=1}^{T_m} \sigma_{mt} a_{mtn} w_{mt}^\circ \tag{15}$$

From these sums form the $\mathbf{N} \times \mathbf{1}$ vector

$$\mathbf{s}_o^\circ = (s_{10}^\circ, \ldots, s_{N0}^\circ)^T \tag{16}$$

(the superscript T denotes transposition) and the $\mathbf{N} \times \mathbf{M}$ matrix

$$S^\circ \equiv \begin{pmatrix} s_{11}^\circ, \ldots, s_{1M}^\circ \\ \cdot \quad\quad \cdot \\ \cdot \quad\quad \cdot \\ \cdot \quad\quad \cdot \\ s_{N1}^\circ, \ldots, s_{NM}^\circ \end{pmatrix} \tag{17}$$

assumed to be of rank M, so that $\mathbf{S}^{\circ T}\,\mathbf{S}^\circ$ is nonsingular. Then compute the $\mathbf{M} \times 1$ vector of multipliers

$$\lambda^\circ \equiv (\mathbf{S}^{\circ T}\,\mathbf{S}^\circ)^{-1}\,\mathbf{S}^{\circ T}\,\mathbf{s}_o^\circ \tag{18}$$

Let the total number of terms be

$$T \equiv \sum_{m=0}^{M} T_m \tag{19}$$

and form the $\mathbf{T} \times \mathbf{1}$ vector

$$\mathbf{w}^\circ \equiv (w_{01}^\circ, \ldots, w_{0T_0}^\circ, w_{11}^\circ, \ldots, w_{1T_1}^\circ, \cdots, w_{M1}^\circ, \ldots, w_{MT_M}^\circ) \tag{20}$$

The vectors $\mathbf{w}^\circ$ and λ° are needed to begin the computation process.

Consider now the i^{th} cycle of the iteration process. The algorithm generates corrections $\Delta\mathbf{w}^i$ and $\Delta\lambda^i$ to the current values $\mathbf{w}^i$ and λ^i of the weights and multipliers, yielding estimates $\mathbf{w}^{i+1}$ and λ^{i+1} for the next iteration.

$$\mathbf{w}^{i+1} \equiv \mathbf{w}^i + \Delta\mathbf{w}^i; \quad \lambda^{i+1} \equiv \lambda^i + \Delta\lambda^i \tag{21}$$

To do this, evaluate sums $\mathbf{s}_{nm}^i$ and form from them a vector $\mathbf{s}_o^i$ and an $\mathbf{N} \times \mathbf{M}$ matrix $\mathbf{S}^i$ just as in Equations (15) to (17), the superscript being i instead of zero. Also compute the $\mathbf{N}^2$ totals for $\mathbf{n}, \mathbf{j} = 1, \ldots, \mathbf{N}$

$$t_{nj}^i \equiv -\sum_{t=1}^{T_0} \sigma_{ot} a_{otn} a_{otj} w_{ot}^i + \sum_{m=1}^{M} \lambda_m^i \sum_{t=1}^{T_m} \sigma_{mt} a_{mtn} a_{mtj} w_{mt}^i \tag{22}$$

and assemble them into an $\mathbf{N}^2$ symmetric matrix

$$\mathbf{T}^i \equiv \begin{pmatrix} t_{11}^i \cdots t_{1N}^i \\ \cdots \\ \cdots \\ \cdots \\ t_{N1}^i \cdots t_{NN}^i \end{pmatrix} \tag{23}$$

At each iteration there is known the value of one additional variable v^i, which is also adjusted by the algorithm. To begin, take it to be the value of the objective function at $\mathbf{x}^\circ$.

$$v^\circ \equiv \sigma g_0^\circ \tag{24}$$

Form the $(\mathbf{M}+\mathbf{N}+1)^2$ symmetric Newton-Raphson matrix $\mathbf{R}^i$

$$\mathbf{R}^i = \begin{pmatrix} \mathbf{T}^i & \mathbf{s}_o^i & \mathbf{S}^i \\ \mathbf{s}_o^{iT} & -1 & \mathbf{0} \\ \mathbf{S}^{iT} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{25}$$

where $\mathbf{0}$ represents null matrices of appropriate dimension to fill out the rows and columns of $\mathbf{R}^i$. Observe that the assumption that $\mathbf{S}^i$ has rank $\mathbf{N}$ implies that $\mathbf{R}^i$ is nonsingular. Next construct the $(\mathbf{M}+\mathbf{N}+\mathbf{1})\times 1$ error vector $\mathbf{e}^i$

$$\mathbf{e}^i = \begin{pmatrix} s_o^i - \mathbf{S}^i\lambda^i \\ \sigma - g_0^i(v^i)^{-1} \\ 1 - g_1^i \\ \vdots \\ 1 - g_m^i \end{pmatrix} = \begin{pmatrix} \mathbf{s}_o^i - \mathbf{S}^i\boldsymbol{\lambda}^i \\ \sigma - g_0^i(v^i)^{-1} \\ 1 - g^i \end{pmatrix} \tag{26}$$

where the third member above merely defines the $\mathbf{M}\times\mathbf{1}$ vectors $\mathbf{1}$ (all components unity) and $\mathbf{g}$ (the vector of constraint function values). Then an $(\mathbf{M}+\mathbf{N}+1)\times 1$ vector of adjustments is given by

$$\begin{pmatrix} \Delta \ln \mathbf{x}^i \\ \Delta \ln v^i \\ \Delta\boldsymbol{\lambda}^i \end{pmatrix} = (\mathbf{R}^i)^{-1}\,\mathbf{e}^i \tag{27}$$

Here $\ln \mathbf{x}^i$ represents the $\mathbf{N}\times 1$ vector of natural logarithms of the $\mathbf{x}_n^i$, so that the next estimate $\mathbf{x}^{i+1}$ is

$$x_n^{i+1} = x_n^i \exp(\Delta \ln x_n^i) \tag{28}$$

whereas

$$v^{i+1} = v^i \exp(\Delta \ln v^i) \tag{29}$$

These quantities are used to compute new values of the weights, defined by Equation (13) for $\mathbf{m}=1,\cdots,\mathbf{M}$, but given for $\mathbf{m}=0$ by

$$w_{ot}^{i+1} \equiv (v^{i+1})^{-1} \quad c_{\sigma t} \prod_{n=1}^{N} x_n^{a_{\sigma t n}} \tag{30}$$

Since the new λ^{i+1} are known from Equations (21) and (27), the algorithm has completed the i^{th} iteration. The procedure continues until all components of the error vector are acceptably close to zero.

SAMPLE CALCULATION

The importance of initial approximations in obtaining the local minimum in as few iterations as possible can be illustrated by the following example: Find the vector $(\mathbf{x}_1^*, \mathbf{x}_2^*) > \mathbf{0}$ that

$$\text{minimizes } g_0 = 5x_1^3 + 10x_2^2 - 50x_1x_2 - 30x_1^2 \tag{31}$$

subject to the constraint

$$g_1 = \frac{x_1^2}{1200} + \frac{x_2^2}{1200} = 1 \tag{32}$$

where $g_0^* < 0$ so that $\sigma = -1$. A local minimum for this problem occurs at $x_1^* = 12.58$ and $x_2^* = 32.27$ where $g_0^* = -4677.57$ and the multiplier λ_1^* has the value -0.064895. Setting $x_1^\circ = 12.6$ and $x_2^\circ = 32.3$, a good approximation to the optimal solution, the problem was solved for different values of λ_1° with the number of iterations shown in Table 2. A poor approximation greatly increases the computational burden, encouraging the use of the algorithm initiation procedures even when a good approximation to the $\mathbf{x}_n^*$ variables is available.

TABLE 2. NUMBER OF ITERATIONS FOR DIFFERENT λ_1°

λ_1°	No. of iterations	λ_1°	No. of iterations
0	3	0	3
0.1	3	−0.1	3
1	4	−1	3
2	5	−2	4
5	7	−5	7
10	13	−10	12

The efficiency of the algorithm for different initial estimates $\mathbf{x}_1^\circ, \mathbf{x}_2^\circ$ when the multiplier λ_1° is obtained by Equation (12) is shown in Table 3. Points 2 and 3 are interior to the circle described by the constraint but quite close to the optimum. Points 4 to 6 are also interior but poor approximations to x_1^*, x_2^*. Finally the number of iterations for points 7 and 8 exterior to the circle are presented. The value of good estimates for large systems will be discussed further in the paper.

TABLE 3. NUMBER OF ITERATIONS FOR DIFFERENT $\mathbf{x}_1^\circ, \mathbf{x}_2^\circ$

Point No.	x_1°	x_2°	λ_1°	g_0°	$1-g_1^\circ$	No. of iterations
1	12.6	33.2	0.128	−4654.5	−0.0508	3
2	10.0	30.0	0.320	−4000.0	0.167	4
3	15.0	25.0	0.093	−2375.0	0.292	5
4	10.0	10.0	0.250	−2000.0	0.833	5
5	1.0	1.0	−0.962	−65.0	0.998	12
6	0.05	0.05	−0.998	−0.2	1.000	100+
7	12.0	35.0	0.381	−4430.0	−0.141	4
8	13.0	45.0	1.87	−3085.0	−0.828	5

To conclude this section, the progress of the algorithm starting from the point $x_1^\circ = x_2^\circ = 10$ is presented in Table 4. Note the quadratic convergence behavior demonstrated in this example.

TABLE 4. SOLUTION FOR $x_1^\circ = 10$, $x_2^\circ = 10$

Iteration No.	$x_1^{(n)}$	$x_2^{(n)}$	$\lambda_1^{(n)}$	$g_0(x_1^{(n)},x_2^{(n)})$	$1-g_1(x_1^{(n)},x_2^{(n)})$
1	10.0	10.0	0.250	−2000.0	0.833
2	13.14	27.18	0.492	−4305.6	0.240
3	12.73	33.29	−0.168	−4653.9	−0.0584
4	12.59	32.30	−0.0738	−4677.0	−0.00170
5	12.58	32.28	−0.0649	−4677.6	-4.14×10^{-5}

JUSTIFICATION

The algorithm is a Newton-Raphson procedure for simultaneously satisfying the primal constraints and driving to zero the gradient of a certain Lagrangian function. Derivations justifying the formulas given previously are presented here.

In reference *1* it is proven that the original primal minimization problem defined in Equations (7) to (13) is equivalent, that is, has the same solution, as the following one:

$$\min_{\mathbf{x}} (\sigma \ln \sigma\, g_0) \tag{33}$$

subject to

$$\ln g_m = 0;\ m = 1, \cdots, M \tag{34}$$

Introduce a Lagrange multiplier (3) λ_m for each of the **M** constraints and form the logarithmic Lagrangian function:

$$L \equiv \sigma \ln \sigma\, g_0 - \sum_{m=1}^{M} \lambda_m \ln g_m \tag{35}$$

It is well known (3) that at a local minimum the first derivatives of **L** must vanish. Let the minimizing values of all variables and functions at a local minimum be marked with *. Then for all $\mathbf{j} = 1, \cdots, \mathbf{N}$

$$\left(\frac{\partial L}{\partial x_j}\right)^* = 0 = (x_j^*)^{-1}\left[(g_0^*)^{-1}\sigma\left\{\sum_{t=0}^{T_0}\sigma_{ot}a_{otj}c_{ot}\prod_{n=1}^{N}(x_n^*)^{a_{otn}}\right\} - \sum_{m=1}^{M}\lambda_m^*(g_m^*)^{-1}\left\{\sum_{t=1}^{T_m}\sigma_{mt}a_{mtn}c_{mt}\prod_{n=1}^{N}(x_n^*)^{a_{mtn}}\right\}\right] \tag{36}$$

This expression is simplified by factoring out x_j^*, which is finite and positive, by Equation (2), and by introducing the following weights:

$$w_{mt} = \begin{cases} v^{-1}\, c_{ot}\prod_{n=1}^{N} x_n^{a_{otn}} & \text{for } m = 0 \\ c_{mt}\prod_{n=1}^{N} x_n^{a_{mtn}} & \text{for } m = 1, \cdots, M \end{cases} \tag{37}$$

Here **v** is regarded as an estimate of σg_0^* generated by the algorithm. The simplified version of Equation (30) is, for $n = 1, \cdots, \mathbf{N}$

$$v^*(g_0^*)^{-1}\sigma\sum_{t=1}^{T_0}\sigma_{ot}a_{otn}w_{ot}^* - \sum_{m=1}^{M}\lambda_m^*(g_m^*)^{-1}\sum_{t=1}^{T_m}\sigma_{mt}a_{mtn}w_{mt}^* = 0 \tag{38}$$

Since $\mathbf{x}^*$ must satisfy the constraint Equation (6)

$$g_m^* = 1\ (m \neq 0) \tag{39}$$

Also **v** must equal σg_0^* at $\mathbf{x}^*$ by definition.

$$v^* = \sigma g_0^* \tag{40}$$

Equations (36) to (38) give

$$\sum_{t=1}^{T_0}\sigma_{ot}a_{otn}w_{ot}^* - \sum_{m=1}^{M}\lambda_m^*\sum_{t=1}^{T_m}\sigma_{mt}a_{mtn}w_{mt}^* = 0 \tag{41}$$

Equations (7), defining the polynomials, and (37), defining the weights, yield, upon summation

$$\sum_{t=1}^{T_0}\sigma_{ot}w_{ot} = g_0 v^{-1} \tag{42}$$

$$\sum_{t=1}^{T_m}\sigma_{mt}w_{mt} = g_m;\ m \neq 0 \tag{43}$$

At a feasible local optimum, Equations (12), (40), (42), and (43) give

$$\sum_{t=1}^{T_0}\sigma_{ot}w_{ot}^* = \sigma \tag{44}$$

$$\sum_{t=1}^{T_m}\sigma_{mt}w_{mt}^* = 1 \qquad m = 1, \cdots, M \tag{45}$$

The algorithm is intended to generate solutions λ^* and $\mathbf{w}^*$ satisfying the **N** orthogonality conditions (42) and the $M + 1$ normality conditions (44), (45).

For the initial estimate $\mathbf{x}^\circ$ it is natural, in view of Equation (34), to take

$$v^\circ = \sigma g_0^\circ \tag{46}$$

This fixes all the weights. Application of abbreviation, Equations (9) to (11), converts Equation (41), written formally for $\mathbf{w}^\circ$ [defined in Equation (13)], and λ° instead of $\mathbf{w}^*$ and λ^*, into

$$\mathbf{S}^\circ\lambda_0^\circ = \mathbf{s}_0^\circ \tag{47}$$

Equation (17) then gives the least squares solution to this inconsistent set of **N** linear equations in only **M** unknowns λ_m. That is, it yields the vector λ of Lagrange multipliers minimizing the sum of squared differences between the left and right members of Equation (47).

It is convenient to rewrite Equation (37) as

$$w_{ot} = \exp\left[\ln c_{ot} + \sum_{n=1}^{N} a_{otn}\ln x_n - \ln(v)\right] \tag{48}$$

and

$$w_{mt} = \exp\left[\ln c_{mt} + \sum_{n=1}^{N} a_{mtn}\ln x_n\right] \text{ for } m \neq 0 \tag{49}$$

Differentiation gives

$$\frac{\partial w_{mt}}{\partial \ln x_j} = a_{mtj}w_{mt} \tag{50}$$

and

$$\frac{\partial w_{mt}}{\partial \ln(v)} = \begin{cases} -w_{ot} & \text{for } m = 0 \\ 0 & \text{for } m \neq 0 \end{cases} \tag{51}$$

Therefore by Equation (22)

$$\frac{\partial}{\partial \ln x_j}\left[\sum_{t=1}^{T_0}\sigma_{ot}a_{otn}w_{ot} - \sum_{m=1}^{M}\lambda_m\sum_{t=1}^{T_m}\sigma_{mt}a_{mtn}w_{mt}\right]^i = \sum_{t=1}^{T_0}\sigma_{ot}a_{otn}a_{otj}w_{ot}^i - \sum_{m=1}^{M}\lambda_m^i\sum_{t=1}^{T_m}\sigma_{mt}a_{mtn}a_{mtj}w_{mt}^i = -t_{nj}^i \quad i, j, n = 1, \cdots, N \tag{52}$$

Moreover, by Equation (15)

$$\frac{\partial}{\partial \ln(v)}\left[\sum_{t=0}^{T_0}\sigma_{ot}a_{otn}w_{ot} - \sum_{m=1}^{M}\lambda_m\sum_{t=1}^{T_m}\sigma_{mt}a_{mtn}w_{mt}\right]^i = -\sum_{t=0}^{T_0}\sigma_{ot}a_{otn}w_{ot}^i = -s_{no}^i \tag{53}$$

Differentiation of the same quantity in Equations (52) and (53) with respect to the λ_m gives, by Equation (15)

$$\frac{\partial[\,]^i}{\partial \lambda_m} = -\sum_{t=1}^{T_m} \sigma_{mt} a_{mtn} w_{mt} = -s^i_{nm} \quad m \neq 0 \tag{54}$$

Hence, to first order, the adjustments needed to annihilate the error in the orthogonality conditions (41) must satisfy the matrix equations

$$\mathbf{T}^i \Delta \ln \mathbf{x}^i + \mathbf{s}_o \Delta \ln v + \mathbf{S}^i \Delta \lambda^i = \mathbf{s}^i_o - \mathbf{S}^i \lambda^i \tag{55}$$

Similarly, by Equations (49) and (15)

$$\frac{\partial \left[\sum_{t=1}^{T_m} \sigma_{mt} w_{mt}\right]^i}{\partial \ln x_n} = \sum_{t=1}^{T_m} \sigma_{mt} a_{mt} w^i_{mt} = s^i_{nm} \tag{56}$$

$$\frac{\partial \left[\sum_{t=1}^{T_m} \sigma_{mt} w_{mt}\right]^i}{\partial \ln v} = \begin{cases} -1 & \text{for} \quad m = 0 \\ 0 & \text{for} \quad m \neq 0 \end{cases} \tag{57}$$

and

$$\frac{\partial \left[\sum_{t=1}^{T_m} \sigma_{mt} w_{mt}\right]^i}{\partial \lambda_m} = 0 \tag{58}$$

whence to first order the corrections must satisfy the scalar equation

$$(\mathbf{s}^i_o)^T \Delta \ln \mathbf{x}_n - \Delta \ln v = \sigma - \sum_{t=1}^{T_0} \sigma_{ot} w^i_{ot} = \sigma - g^i_0 (v^i)^{-1} \tag{59}$$

and the matrix equation

$$(\mathbf{S}^i)^T \Delta \ln \mathbf{x}_n = \mathbf{1} - \mathbf{g}^i \tag{60}$$

The Newton-Raphson matrix $\mathbf{R}^i$ defined in Equation (25) must therefore be inverted to give the adjustments satisfying Equations (55), (59), and (60) with the error vector as defined in Equation (26).

As with any Newton-Raphson method, convergence is not guaranteed. Moreover, even when the algorithm does converge, the point found may be a saddlepoint or even a maximum, since vanishing of the Lagrangian's gradient is only a necessary not a sufficient condition for a local minimum. Hence the second derivatives of any point computed by this algorithm should be checked (3). Finally, a local minimum found may not be the global minimum. These annoyances are characteristic of many optimization techniques and should not preclude successful application of this algorithm to design problems, which often seem well enough behaved in practice.

REACTOR DESIGN PROBLEM SOLUTION

An ALGOL-60 coding of the algorithm was used to solve the reactor design problem expressed here as a generalized polynomial program. Simple order of magnitude engineering estimates of the parameters were used to initiate the algorithm. In many engineering applications, however, the preliminary material and energy balances necessary to verify the model yield not only good estimates of the parameters but also values satisfying the nonlinear constraints. The optimal design shown in Table 5 was obtained in ten iterations using 15-sec. processor time on a B5500 computer. Convergence to the same solution from different starting points decreased the possibility of the existence of false optima. Calculations in the nonisothermal case and for various economic criteria can be found in reference 6.

TABLE 5. OPTIMAL REACTOR DESIGN

Design variable	Initial estimate	Optimal value
A	1.0×10^2	749
c_A	1.0×10^{-1}	0.124
d_V	1.0×10^1	6.65
I	1.0×10^5	92,700
q	1.0×10^3	187
W	1.0×10^3	5,160
w_o	1.0×10^5	86,000
w_i	1.0×10^5	171,000
ΔT	1.0×10^1	29.1

CONCLUSIONS

This report has presented an algorithm for solving generalized polynomial optimization problems. The algorithm, initiated by estimates of the optimal variables, searches for the stationary point of a logarithmic Lagrangian function and relates this point to a local minima of the generalized polynomial problem. When false optima are suspected, great care must be taken to ensure achievement of the desired minima. Computational experience gained working with the algorithm on an engineering design problem with nine variables and six equality constraints has been very promising. The algorithm should be applicable to even larger problems, provided they are well formulated and good initial variable estimates are available.

ACKNOWLEDGMENT

This research was supported by Office of Saline Water Grant 14-01-0001-699.

LITERATURE CITED

1. Passy, U., and D. J. Wilde, *SIAM J. Appl. Math.*, **15** (5), 1344-1356 (1967).
2. Kuhn, H. W., and A. W. Tucker, *Proc. Second Berkeley Symp. Math. Statistics Probability 1950*, **11**, 481-492, Univ. California Press, Berkeley (1951).
3. Wilde, D. J., and C. S. Beightler, "Foundations of Optimization," Prentice-Hall, Englewood Cliffs, N. J. (1967).
4. Duffin, R. J., E. L. Peterson, and C. Zener, "Geometric Programming," Wiley, New York (1966).
5. Avriel, M., and D. J. Wilde, *Ind. Eng. Process Design Develop.*, **6** (2), 256-263 (1967).
6. Blau, G. E. and D. J. Wilde, *Can. J. Chem. Eng.*, **47**, 317-326 (1969).
7. Peters, M. S., and K. D. Timmerhaus, "Plant Design and Economics for Chemical Engineers," 2nd ed, McGraw-Hill, New York (1968).
8. Eben, C. D., and J. R. Ferron, private communication (April 1968).
9. Hildebrand, F. B., "Introduction to Numerical Analysis," p. 443 McGraw-Hill, New York (1956).
10. Westbrook, G. T., and Rutherford, Aris, *Ind. Eng. Chem.*, **53** (3), 181 (1961).
11. Blau, G. E., Ph.D. thesis, Stanford Univ., Calif. (1968).

Manuscript received April 24, 1969; revision received July 28, 1969; paper accepted August 4, 1969. Paper presented at AIChE Cleveland meeting.