Finding Multiple Roots of Nonlinear Algebraic Equations Using S-System Methodology

Michael A. Savageau

Department of Microbiology and Immunology
The University of Michigan Medical School
Ann Arbor, Michigan 48109-0620

Transmitted by Robert Rosen

ABSTRACT

The solution of nonlinear algebraic equations is a well-known problem in many fields of science and engineering. Several types of numerical methods exist, each with their advantages and disadvantages. S-system methodology provides a novel approach to this problem. The result, as judged by extensive numerical tests, is an effective method for finding the positive real roots of nonlinear algebraic equations. The motivation for this method begins with the observation that any system of nonlinear equations composed of elementary functions can be recast in a canonical nonlinear form known as an S-system. When the derivatives of these ordinary first-order nonlinear differential equations are zero, the resulting nonlinear algebraic equations can be represented as an underdetermined set of linear algebraic equations in the logarithms of the variables. These linear equations, together with a set of simple nonlinear constraints, determine the multiple steady-state solutions of the original system that are positive real. The numerical solution is obtained by approximating the constraints as S-system equations in steady state to yield a determined set of linear algebraic equations, which then can be solved iteratively. This method has been implemented and the experience to date suggests that the method is robust and efficient. The rate of convergence to the solutions is quadratic. A combinatorial method for selecting initial conditions has led to the identification of several, and in many cases all, of the positive real solutions for the set of problems tested. It does so without resorting to analysis in the complex domain, and without having to make random or problem-specific provisions for initializing the procedure. There is an obvious parallel implementation of the algorithm, and there are additional generalizations and efficiencies yet to be realized.

1. INTRODUCTION

In this paper I describe a new algorithm for finding multiple roots to equations of the following form

$$f_i(x_1, x_2, \ldots, x_n) = 0 \qquad i = 1, 2, \ldots, n. \tag{1}$$

This is a difficult but commonly encountered problem. Some typical applications include finding roots of polynomials, roots of rational functions, roots of nonlinear algebraic equations, solution of steady-states for dynamic systems, solution of optimization problems, solution of chemical equilibrium equations (multinomial systems), solution of generalized mass-action equations. There are several well-known approaches to this problem; bisection, Newton, and continuation may be considered representative. Each approach has its advantages and disadvantages. A number of reviews can be found in the literature (e.g., [1, 3, 8]).

In exploring methods to solve for the steady states of nonlinear equations that have been recast in a canonical form (see below), I have discovered a simple algorithm for finding the positive real roots of nonlinear algebraic equations. The purpose of this communication is to provide motivation for this approach, to describe the method, and to present a sampling of numerical results. The method has a number of desirable properties. Most notable are (i) it provides an upper bound on the number of positive real solutions, (ii) empirical evidence suggests that it finds several, and in many cases all, of the positive real solutions, (iii) it converges rapidly to these solutions, (iv) it allows solutions to be obtained in parallel, and (v) it can be extended to deal with a broad class of nonlinearities (although the generalized-mass-action form will be the focus of this report).

2. MOTIVATION FOR THE S-SYSTEM APPROACH

Rather arbitrary systems of nonlinear functions can be recast in the following canonical nonlinear form known as an S-system (for a recent review, see Voit [14]):

$$dX_i/dt = \alpha_i \prod_{j=1}^{n} X_j^{g_{ij}} - \beta_i \prod_{j=1}^{n} X_j^{h_{ij}} \qquad X_i(0) = X_{i0} \qquad i = 1, \ldots, n \tag{2}$$

where n is the number of dependent variables, the variables X_i and the multiplicative parameters α_i and β_i are nonnegative real, and the exponential parameters g_{ij} and h_{ij} are real. The existence of such a canonical form suggests that methods developed to deal with S-systems will apply to a broad class of nonlinear problems. For example, a method based on this canonical form has been developed to solve initial value problems, and a diverse set of benchmark tests have shown that it is fast, accurate, predictable and robust [4]. The structural regularity of this canonical form suggests that it might also yield methods for finding the roots of rather general nonlinear algebraic equations.

Furthermore, the steady-state solution of an S-system reduces to a linear algebraic problem in a logarithmic coordinate system [11]. The result is given by

$$y] = [A]^{-1}b] \tag{3}$$

where

$$a_{ij} = g_{ij} - h_{ij}$$
$$y_k = \ln X_k$$
$$b_k = \ln(\beta_k/\alpha_k).$$

When the linear system is underdetermined, the solution is obtained from the set of linear equations with maximum rank together with a set of nonlinear constraints that are generated in the recasting procedure.

3. SOLUTION BY REDUCTIVE RECASTING

The final step in recasting a set of nonlinear equations into S-system form involves reducing multiple sums and differences of terms that involve products of power-law functions to a single difference of terms that involve products of power-law functions [12].

3.1. *One Sign with Multiple Terms*

Consider the following simple example for which the variable X_1 is assumed to approach a steady-state value with time.

$$dX_1/dt = \alpha_{11}X_1^{g_{111}} + \alpha_{12}X_1^{g_{112}} - \beta_1 X_1^{h_{11}} \qquad X_1(0) = X_{10} \tag{4}$$

This equation can be recast to S-system form by letting $X_1 = X_a X_b$ and differentiating the product.

$$X_a dX_b/dt + X_b dX_a/dt = \alpha_{11}X_a^{g_{111}}X_b^{g_{111}} + \alpha_{12}X_a^{g_{112}}X_b^{g_{112}} - \beta_1 X_a^{h_{11}}X_b^{h_{11}} \tag{5}$$

The reductive step involves associating appropriate terms to yield two equations in S-system form.

$$dX_a/dt = \alpha_{11} X_a^{g_{111}} X_b^{g_{111}-1} - \beta_1 X_a^{h_{11}} X_b^{h_{11}-1} \qquad X_a(0) = X_{10} \tag{6}$$

$$dX_b/dt = \alpha_{12} X_a^{g_{112}-1} X_b^{g_{112}} \qquad X_b(0) = 1 \tag{7}$$

From this result, it is immediately apparent that X_a and X_b will not exhibit a steady state even though the original variable X_1 does. The variable X_b increases continually with time because the right-hand side of its equation is positive definite. The variable X_a decreases continually with time in such a manner that the product $X_a X_b$ approaches the steady state of X_1.

This awkwardness is due to the manner in which the negative term was distributed entirely to the equation for X_a. Since the manner of distribution is arbitrary, one can choose the distribution that will produce a true steady state for each of the new variables X_a and X_b. For example,

$$dX_a/dt = \alpha_{11} X_a^{g_{111}} X_b^{g_{111}-1} - \beta_1 X_2 X_a^{h_{11}} X_b^{h_{11}-1} \qquad X_a(0) = X_{10} \tag{8}$$

$$dX_b/dt = \alpha_{12} X_a^{g_{112}-1} X_b^{g_{112}} - \beta_1 X_3 X_a^{h_{11}-1} X_b^{h_{11}} \qquad X_b(0) = 1 \tag{9}$$

where

$$X_2 + X_3 = 1. \tag{10}$$

Equation (10) can be approximated by a product of power-law functions as [10]

$$\gamma_1 X_2^{f_2} X_3^{f_3} = 1 \tag{11}$$

where

$$f_2 = \frac{X_2}{X_2 + X_3} \qquad f_3 = \frac{X_3}{X_2 + X_3} \tag{12}$$

$$\ln(1/\gamma_1) = \frac{X_2}{X_2 + X_3} \ln\left(\frac{X_2}{X_2 + X_3}\right) + \frac{X_3}{X_2 + X_3} \ln\left(\frac{X_3}{X_2 + X_3}\right) \tag{13}$$

and in steady state the set of equations (8), (9), and (11) can be written in linear form as

$$\begin{bmatrix} (g_{111}-h_{11}) & -1 & 0 \\ (g_{112}-h_{11}) & 0 & -1 \\ 0 & f_2 & f_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \ln(\beta_1/\alpha_{11}) \\ \ln(\beta_1/\alpha_{12}) \\ \ln(1/\gamma_1) \end{bmatrix}. \tag{14}$$

The variable elements in the matrix $[A]$ and the vector $b]$ of (14) can be determined from an arbitrary distribution over the auxiliary variables X_2 and X_3. This allows a solution of the linear system, including new estimates for the auxiliary variables, and the procedure is repeated. This iterative algorithm converges rapidly to a distribution over the auxiliary variables that yields a true steady state for the S-system equations produced in the reductive step of recasting. It also yields a steady-state solution for the original variable X_1.

3.2. *Both Signs with Multiple Terms*

Consider the following equation for X_1, which is assumed to approach a steady-state value with time.

$$dX_1/dt = \alpha_{11}X_1^{g_{111}} + \alpha_{12}X_1^{g_{112}} - \beta_{11}X_1^{h_{111}} - \beta_{12}X_1^{h_{112}} \qquad X_1(0) = X_{10} \tag{15}$$

This equation can be recast to S-system form again by letting the original variable be replaced by the product of new variables, the minimum number being two. However, four new variables are needed to ensure that the resulting S-system equations have a true steady state. Let $x_1 = x_a x_b x_c x_d$, differentiate the product, and associate appropriate terms to yield four equations in S-system form.

$$\begin{aligned} dX_a/dt &= \alpha_{11}X_4X_a^{g_{111}}X_b^{g_{111}-1}X_c^{g_{111}-1}X_d^{g_{111}-1} \\ &\quad -\beta_{11}X_2X_a^{h_{111}}X_b^{h_{111}-1}X_c^{h_{111}-1}X_d^{h_{111}-1} \qquad X_a(0) = X_{10} \quad (16) \\ dX_b/dt &= \alpha_{11}X_5X_a^{g_{111}-1}X_b^{g_{111}}X_c^{g_{111}-1}X_d^{g_{111}-1} \\ &\quad -\beta_{12}X_2X_a^{h_{112}-1}X_b^{h_{112}}X_c^{h_{112}-1}X_d^{h_{112}-1} \qquad X_b(0) = 1 \quad (17) \\ dX_c/dt &= \alpha_{12}X_4X_a^{g_{112}-1}X_b^{g_{112}-1}X_c^{g_{112}}X_d^{g_{112}-1} \\ &\quad -\beta_{11}X_3X_a^{h_{111}-1}X_b^{h_{111}-1}X_c^{h_{111}}X_d^{h_{111}-1} \qquad X_c(0) = 1 \quad (18) \\ dX_d/dt &= \alpha_{12}X_5X_a^{g_{112}-1}X_b^{g_{112}-1}X_c^{g_{112}-1}X_d^{g_{112}} \\ &\quad -\beta_{12}X_3X_a^{h_{112}-1}X_b^{h_{112}-1}X_c^{h_{112}-1}X_d^{h_{112}} \qquad X_d(0) = 1 \quad (19) \end{aligned}$$

where

$$X_2 + X_3 = 1 \tag{20}$$

$$X_4 + X_5 = 1. \tag{21}$$

Equations (20) and (21) can be approximated by a product of power-law functions as indicated above

$$\gamma_1 X_2^{f_2} X_3^{f_3} = 1 \tag{22}$$

$$\gamma_2 X_4^{f_4} X_5^{f_5} = 1 \tag{23}$$

where

$$f_2 = \frac{X_2}{X_2 + X_3} \qquad f_3 = \frac{X_3}{X_2 + X_3} \tag{24}$$

$$\ln(1/\gamma_1) = \frac{X_2}{X_2 + X_3} \ln\left(\frac{X_2}{X_2 + X_3}\right) + \frac{X_3}{X_2 + X_3} \ln\left(\frac{X_3}{X_2 + X_3}\right) \tag{25}$$

$$f_4 = \frac{X_4}{X_4 + X_5} \qquad f_5 = \frac{X_5}{X_4 + X_5} \tag{26}$$

$$\ln(1/\gamma_2) = \frac{X_4}{X_4 + X_5} \ln\left(\frac{X_4}{X_4 + X_5}\right) + \frac{X_5}{X_4 + X_5} \ln\left(\frac{X_5}{X_4 + X_5}\right) \tag{27}$$

and in steady state, the set of equations (16)–(18), (22)–(23) can be written in linear form as

$$\begin{bmatrix} (g_{111}\text{-}h_{111}) & -1 & 0 & 1 & 0 \\ (g_{111}\text{-}h_{112}) & -1 & 0 & 0 & 1 \\ (g_{112}\text{-}h_{111}) & 0 & -1 & 1 & 0 \\ 0 & f_2 & f_3 & 0 & 0 \\ 0 & 0 & 0 & f_4 & f_5 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} \ln(\beta_{11}/\alpha_{11}) \\ \ln(\beta_{12}/\alpha_{11}) \\ \ln(\beta_{11}/\alpha_{12}) \\ \ln(1/\gamma_1) \\ \ln(1/\gamma_2) \end{bmatrix}. \tag{28}$$

Note that (19) is a linear combination of (16)–(18) in steady state, and so has been eliminated.

The variable elements in the matrix $[A]$ and the vector $b]$ of (28) can be determined from an arbitrary distribution over the auxiliary variables X_2 and X_3, and X_4 and X_5. This allows a solution of the linear system, including new estimates for the auxiliary variables, and the procedure is repeated. This iterative algorithm converges rapidly to a distribution over the auxiliary variables that yields a true steady state for the S-system equations produced in the reductive step of recasting. It also yields a steady-state solution for the original variable X_1.

3.3. *Generalization*

The algorithm described above in the context of two simple examples is readily generalized to treat systems of nonlinear algebraic equations having an arbitrary number of equations (variables) with an arbitrary number of positive and negative terms in each equation. The auxiliary variables exist as disjoint sets numbering at most two for each of the original equations. The number of auxiliary variables in each of these sets is determined by the number of terms of a given sign in each of the original equations.

If one assigns an initial value of unity to one of the auxiliary variables in each set and zero to all the others in each set, then the number of different initial assignments is

$$\prod_{k=1}^{NAS} NAV(k) \tag{29}$$

where

$$NAS = \text{Number of Auxiliary Sets}$$

$$NAV(k) = \text{Number of Auxiliary Variables in the } k\text{th set.}$$

This product also provides an upper bound on the number of positive real roots. This bound is less than Bezout's number [1, 8] in the case of multinomial equations with a high order but few terms per equation; it is greater than Bezout's number in the case of multinomial equations with low order but many terms per equation. Yet another upper bound is less than either of these in some instances. It is calculated as follows. First, determine the minimum number of terms of a given sign for each equation. Second, double these numbers and subtract one from each doubled number that is greater than two. Finally, take the product of the resulting numbers to give an upper bound on the number of positive real roots. The search for additional roots can be terminated once any of these bounds is reached.

The algorithm as currently implemented has found several, and in many cases all, of the positive real roots for the problems tested when one chooses the initial values for the auxiliary variables according to the combinatorial scheme described above. The method can be generalized in a number of ways, as will be described below.

4. NUMERICAL EXAMPLES

The following examples are representative of the test problems that have been examined. In each case a tolerance parameter has been set at 1.0e-8.

EXAMPLE 1. The stationary points of Himmelblau's function [9] are given by the roots of the following system of equations, which exhibit both positive and negative real roots.

$$2x_1^3 + 2x_1x_2 + x_2^2 - 21x_1 - 7 = 0 \tag{30}$$

$$x_1^2 + 2x_1x_2 + 2x_2^3 - 13x_2 - 11 = 0 \tag{31}$$

The three positive real roots are found as in Table 1. These require five to eight iterations and the rate of convergence is approximately quadratic. The

TABLE 1
ROOTS WITH NEGATIVE AS WELL AS POSITIVE REAL COMPONENTS FOR THE PROBLEM IN EXAMPLE 1

		Rate of Convergence[†]	
Roots*	Iterations	Average[‡]	Final[§]
{3.000e + 00, 2.000e + 00}	5	2.33	2.04
{3.385e + 00, 7.385e − 02}	8	1.97	1.91
{8.668e − 02, 2.884e + 00}	6	2.19	1.98
{−2.805e + 00, 3.131e + 00}	6	3.08	2.21
{3.584e + 00, −1.848e + 00}	6	2.68	2.01
{−1.280e − 01, −1.954e + 00}	5	2.29	1.97
{−2.708e − 01, −9.230e − 01}	6	3.26	2.08
{−3.073e + 00, −8.135e − 02}	6	2.04	1.92
{−3.779e + 00, −3.283e + 00}	5	3.16	2.31

* The tolerance was set at e − 8, but the data are reported with four significant digits.

[†] Rate of convergence between two iterations is calculated as $r_i = \log(e_i/e_{i-1})$.

[‡] Average is the geometric mean calculated from the iteration that first produces an error less than 1.

[§] Rate of convergence between the last two iterations before the tolerance condition is met.

method also finds roots with a negative real component. These are obtained by replacing x_i by $-x_i$, which leads to a reversal of sign for odd functions of x_i, and solving for the positive real roots of the resulting equations. This yields the six roots with a negative real component (Table 1). A similar number of iterations and a similar rate of convergence are observed for these solutions. Thus, the method finds nine real solutions, which is the total for this set of equations [6].

EXAMPLE 2. The following system of equations was constructed to test the ability of the method to identify roots with one or more components having a value of zero.

$$3x_1^{1.1}x_2^{2.06} + x_1^3x_2^6 + x_3 - 10x_1^2x_2^4 - x_4 = 0 \tag{32}$$

$$10x_1^{2.03}x_2 + x_1^5x_2^{2.7} + x_3 - 30x_1^4x_2^2 - x_4 = 0 \tag{33}$$

$$0.02x_3 - 0.01x_3^2 = 0 \tag{34}$$

$$0.02x_4 - 0.01x_4^2 = 0 \tag{35}$$

The four positive real roots are found as in Table 2. In this example, three to six iterations are required, and the rate of convergence is again approximately quadratic. The method also finds all the roots with one or more components equal to zero (Table 2). In some of these cases, the equations reduce exactly to S-systems and the solution is obtained directly without iteration. Otherwise three to seven iterations are required, and the rate of convergence is nearly quadratic. The method finds all of the nonnegative real roots.

EXAMPLE 3. The following is a well-known problem that is often used to test methods of solving systems of nonlinear equations [7]. It is a chemical equilibrium problem, and therefore all components of the solution must be positive real. Furthermore, it is known from thermodynamic principles that there can be only one such solution [5].

$$x_1x_2 + x_1 - 3x_5 = 0 \tag{36}$$

$$\begin{aligned}2x_1x_2 + x_1 + 1.923e{-6}\ x_2^2 + x_2x_3^2 + 5.45177e{-4}\ x_2x_3&\\ + 3.40735e{-5}\ x_2x_4 + 4.4975e{-7}x_2 - 10x_5 = 0&\end{aligned} \tag{37}$$

$$2x_2x_3^2 + 5.45177e{-4}\ x_2x_3 + 3.86e{-1}\ x_3^2 + 4.10622e{-4}x_3 - 8x_5 = 0 \tag{38}$$

$$3.40735e{-5}\ x_2x_4 + 2x_4^2 - 40\ x_5 = 0 \tag{39}$$

TABLE 2

ROOTS WITH ZERO AS WELL AS POSITIVE REAL COMPONENTS FOR THE PROBLEM IN EXAMPLE 2

		Rate of Convergence[†]	
Roots*	Iterations	Average[‡]	Final[§]
$\{6.803e-01, 6.526e-01, 2.000e+00, 2.000e+00\}$	6	2.02	1.95
$\{5.547e+01, 4.150e-01, 2.000e+00, 2.000e+00\}$	3	2.52	2.27
$\{2.856e+02, 3.997e-02, 2.000e+00, 2.000e+00\}$	4	2.04	2.03
$\{1.582e-01, 7.834e+00, 2.000e+00, 2.000e+00\}$	5	1.94	1.96
$\{0.000e+00, 0.000e+00, 2.000e+00, 2.000e+00\}$	1	—	—
$\{5.540e+01, 4.158e-01, 0.000e+00, 2.000e+00\}$	7	2.49	2.26
$\{5.555e+01, 4.142e-01, 2.000e+00, 0.000e+00\}$	3	2.49	2.25
$\{2.547e-01, 6.163e+00, 2.000e+00, 0.000e+00\}$	4	4.79	2.23
$\{6.855e-01, 9.673e-01, 2.000e+00, 0.000e+00\}$	4	2.63	2.32
$\{2.176e+02, 5.892e-02, 2.000e+00, 0.000e+00\}$	5	2.28	2.09
$\{6.803e-01, 6.526e-01, 0.000e+00, 0.000e+00\}$	3	2.70	2.31
$\{1.582e-01, 7.834e+00, 0.000e+00, 0.000e+00\}$	3	2.75	2.32
$\{2.856e+02, 3.997e-02, 0.000e+00, 0.000e+00\}$	3	2.50	2.26
$\{5.547e+01, 4.150e-01, 0.000e+00, 0.000e+00\}$	3	2.50	2.26
$\{0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00\}$	1	—	—

* The tolerance was set at e − 8, but the data are reported with four significant digits.

[†] Rate of convergence between two iterations is calculated as $r_i = \log(e_i/e_i)$.

[‡] Average is the geometric mean calculated from the iteration that first produces an error less than 1.

[§] Rate of convergence between the last two iterations before the tolerance condition is met.

$$x_1x_2 + x_1 + 9.615e{-}7x_2^2 + x_2x_3^2 + 5.45177e{-}4\ x_2x_3 + 3.40735e{-}5\ x_2x_4 + 4.4975e{-}7\ x_2 + 1.930e{-}1\ x_3^2 + 4.10622e{-}4\ x_3 + x_4^2 - 1 = 0 \tag{40}$$

The one physically realizable solution in this case is $\{3.114102e-03, 3.459792e+01, 6.504177e-02, 8.593780e-01, 3.695185e-02\}$. This is the only solution obtained; it is obtained repeatedly with an average of nine iterations. The averages for these repeated solutions are the following: the average rate of convergence is 1.85, and the final rate of convergence is 1.92. This problem exhibits the major drawback of the current implementation; namely, the same solution is obtained repeatedly. Of course, when one knows that for physical reasons there can only be a single solution, as in the case of chemical equilibrium, one could stop the procedure after obtaining the first solution.

5. DISCUSSION

The method, as currently implemented, finds nonnegative real roots for systems of generalized-mass-action equations (generalized polynomials)

$$\sum_{k=1}^{p} \alpha_{ik} \prod_{j=1}^{n} X_j^{g_{ijk}} - \sum_{k=1}^{p} \beta_{ik} \prod_{j=1}^{n} X_j^{h_{ijk}} = 0 \qquad i = 1, 2, \ldots, n \tag{41}$$

where n is the number of equations and variables, p is the maximum number of terms of a given sign in each equation, g_{ijk} and h_{ijk} are real, α_{ik} and β_{ik} are nonnegative real, and X_j are nonnegative real.

In the special case (multinomial systems) where g_{ijk} and h_{ijk} are integers, the method also finds negative real roots (Example 1). Complex roots can be obtained by allowing the multiplicative parameters and variables to have real and imaginary parts, expanding the resulting equations, collecting real and imaginary parts, setting each part equal to zero, and then applying the method to this expanded set of equations as before. I have found the complex roots for a few simple test problems by performing the expansion manually and applying the method. Thus, this method also can be used to find complex roots.

The current implementation has two principal limitations. In some cases, it misses specific roots and in others it finds some roots repeatedly. To date, the method has found all of the positive real roots for approximately 100 test problems. However, there also are a few (and perhaps many more) problems for which it finds several but not all of the positive real roots. It remains to be determined whether or not the method can be modified so as to guarantee that it will find all of the positive real roots. On the other hand, the method typically finds some solutions repeatedly. The method provides information that can be used to suppress this redundancy, although at present it does not completely eliminate it. This also is a key area for further development.

The current implementation also exhibits a number of desirable properties. The rate of convergence is typically quadratic, even when far from a solution. We are exploring methods to accelerate this rate. The speed of solution also is dependent on the size of the system. The method tends to generate arrays that become increasingly sparse as the number of variables increases, e.g., compare (14) and (28). Thus, a new implementation that deals efficiently with these sparse arrays will improve efficiency of both memory management and speed. Increased speed also could be obtained by means of an obvious parallel implementation based on the combinatorial selection of initial values. Each of M machines can be initialized to seek a different solution, and then they can be run independently. Finally, the method can be applied to systems of more general nonlinear functions than those described in this report. Recasting such equations as S-systems and setting the derivatives to zero leads in general to a set of underdetermined linear equations in logarithmic coordinates. In order to solve these equations one must add the nonlinear equations that define the recast variables [12]. Decomposing these nonlinear constraints into disjoint sets of convex constraint relationships, as in the case of the auxiliary variables above, will reduce these problems to a form that the method already can solve efficiently.

Generalized-mass-action equations in addition to being recast exactly as S-systems, as described above, also can be represented locally by S-system equations [13]. This is true of other nonlinear equations as well. Voit and Irvine [15] have solved iteratively for the steady states of such equations by using Newton's method in a logarithmic space although only tentative implementations of this approach were developed. Burns and Locascio [2] have recently developed a method that involves the same principles; namely, aggregating terms with the same sign, locally approximating each aggregate about the current values by a product of power-law functions, converting to a linear system, and solving the system iteratively. These authors have characterized the basins of attraction for the various roots and have shown that the method is often more robust than is the conventional Newton method. Indeed, they have shown that this method has a number of desirable properties, including invariance to various features of problem representation. Their method is different but obviously related to the method described in this paper. Comparisons of these methods are now in progress.

REFERENCES

1 E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*, Vol. 13; Series in Computational Mathematics, Springer-Verlag, Berlin, New York, 1990.

2 S. A. Burns and A. Locascio, A monomial-based method for solving systems of non-linear algebraic equations, *Internat. J. Numer. Methods. Engrg.* 31:1295–1318 (1991).

3 C. B. Garcia and W. I. Zangwill, *Pathways to Solutions, Fixed Points, and Equilibria*, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

4 D. H. Irvine and M. A. Savageau, Efficient solution of nonlinear ordinary differential equations expressed in S-system canonical form, *SIAM J. Numer. Anal.* 27:704–735 (1990).

5 H. J. Kandiner and S. R. Brinkley, Calculation of complex equilibrium relations, *Industrial and Engineering Chemistry* 42:850–855 (1950).

6 W. J. Lin, J. D. Seader, and T. L. Wayburn, Computing multiple solutions to systems of interlinked separation columns, *AIChE J.* 33:886–897 (1987).

7 K. Meintjes and A. P. Morgan, Chemical equilibrium systems as numerical test problems, *ACM Trans. Math. Software* 16:143–151 (1990).

8 A. P. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

9 G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell, *Engineering Optimization*, Wiley, New York, 1983, p. 69.

10 M. A. Savageau, *Biochemical Systems Analysis: A Study of Function and Design in Molecular Biology*, Addison-Wesley, Reading, Massachusetts, 1976.

11 M. A. Savageau, Biochemical systems analysis II. The steady state solutions for an n-pool system using a power-law approximation, *J. Theoret. Biol.* 25:370–379 (1969).

12 M. A. Savageau and E. O. Voit, Recasting nonlinear differential equations as S-systems: A canonical nonlinear form, *Math. Biosci.* 87:83–115 (1987).

13 A. Sorribas and M. A. Savageau, A comparison of variant theories of intact biochemical systems II: Flux-oriented and metabolic control theories, *Math. Biosci.* 94:195–238 (1989).

14 E. O. Voit, *Canonical Nonlinear Modeling: S-System Approach to Understanding Complexity*, Van Nostrand Reinhold, New York, 1991.

15 E. O. Voit and D. H. Irvine (unpublished results).