Computers & Geosciences Vol. 16, No. 1, pp. 21–40, 1990
Printed in Great Britain.

0098-3004/90 $3.00 + 0.00

A PENALTY FUNCTION METHOD FOR COMPUTING CHEMICAL EQUILIBRIA

DAVID J. KIRKNER
Department of Civil Engineering, University of Notre Dame, Notre Dame, IN 46556, U.S.A.

and

HOWARD W. REEVES
Department of Civil Engineering, University of Michigan, Ann Arbor, MI 48109, U.S.A.

(*Received* 17 *August* 1987; *revised* 10 *August* 1988; *received for publication* 20 *June* 1989)

Abstract—A penalty function technique is developed as an alternative method for handling precipitation–dissolution reactions in the equilibrium constant method of solving the batch chemical equilibrium problem. It is simple to incorporate into existing programs which do not handle precipitation–dissolution reactions. No variable eliminations need to be performed and the size of the system is always the same even as additional solids enter the problem. Two sample problems are presented to demonstrate the implementation of the method.

Key Words: Chemical equilibrium, Computation, Mathematics, Penalty function, Precipitation/dissolution.

INTRODUCTION

There are many problems in geochemistry and geophysics where knowledge of the equilibrium state of a set of reacting chemical constituents is important. Since the 1940s and the pioneering work of Brinkley (1946, 1947) much progress has been made in developing efficient computational procedures to solve this problem with a variety of different applications in mind. A good review of different approaches and existing software for equilibrium calculations in aqueous systems is given by Nordstrom and others (1979).

There are two basic approaches to the equilibrium problem: the Gibb's energy minimization and the so-called equilibrium constant approach. For fairly large complex chemical systems, of interest in geochemical applications, the latter approach is desirable primarily because of the lack of thermodynamic data required for the former. This paper is concerned with only the equilibrium constant method. As a perusal of Nordstrom and others (1979) indicates, there are many available programs based on the equilibrium constant method. One reason that many programs have been and continue to be developed is that the database (the set of chemical constants which characterize the equilibrium state) required for large complex chemical systems is large and different applications require different databases.

Existing programs use various methods (of concern here is principally the Newton–Raphson method) to solve the nonlinear algebraic equations that govern the equilibrium calculation. The approaches differ in efficiency. However it could be argued that even for a large chemical system (e.g. 100 components) the number of equations involved do not pose much of a burden on modern computers. Compare, for instance, the computational effort required for the approximate solution of sets of nonlinear partial differential equations which may involve solving thousands of nonlinear algebraic equations. Thus if a program for equilibrium chemical calculations works, the relative efficiency might not seem important. However, a fairly new application has arisen in recent years which requires programs which are flexible and efficient. This application is modeling the transport of reacting solutes through porous media. Several general approaches to this problem have been advanced in the literature recently; for example Walsh and others (1984) and Kirkner, Theis, and Jennings (1984). Although each of these works solve the governing equations with a different algorithm, they possess the common feature of iterating between two sets of equations; one set contains in essence the discretized transport equations, and the second set contains the algebraic equations of chemical equilibrium at each nodal point in the domain. Thus in a typical simulation chemical equilibrium calculations may be performed at hundreds of iterations and time steps. The efficiency of the chemical equilibrium calculations is thus of considerable importance.

For systems involving only an aqueous phase the formulation of the nonlinear algebraic equations is standard and solution by Newton–Raphson iteration straightforward. Variants of this method and other nonlinear algebraic equation algorithms also have been employed (Morin, 1985). If a solid phase is present resulting from a precipitation–dissolution reaction, there are several approaches which have

been employed in the literature. These different approaches will be reviewed herein and a new method utilizing a penalty function approach introduced. This modified penalty technique offers some insight into the solubility product as a constraint on the system and is simple to implement. Two sample calculations are presented to clarify the implementation of the method. This paper is basically an exposition of the method; much development work is required yet before its relative efficiency can be evaluated properly for large chemical systems.

For simplicity in presentation ideality is assumed in the following, that is concentrations of aqueous species are taken equal to activities. The algorithm presented is modified easily to use activity coefficients, however their inclusion in the presentation obscures the basic ideas.

CHEMICAL EQUILIBRIUM IN HOMOGENEOUS AQUEOUS SYSTEMS

As originally developed by Brinkley (1946), N constituents of a homogeneous aqueous chemical system may be divided into N_c components and N_{cx} complexes. The number of components of a system is defined as the smallest number of substances required to determine the concentration of all species in the system. A component itself is a species in the system. Complexes are considered products in reactions where the components are the reactants. These reactions can be written

$$\sum_{j=1}^{N_c} A_{ij}\hat{c}_j \rightleftharpoons \hat{x}_i \ (i = 1,2, \ldots, N_{cx}). \quad (1)$$

A specific example of these reactions is given for the $CaCO_3$ system in Appendix 2. This sample problem is used later for some numerical results. In Equation (1) $\hat{c}_j$ (respectively $\hat{x}_i$) represents the chemical formula for component j (respectively complex i). The A_{ij} are the stoichiometric coefficients. The mathematical statement of equilibrium for each reaction in (1) is the law of mass action

$$x_i = K_i \prod_{j=1}^{N_c} c_j^{A_{ij}} \ (i = 1, \ldots, N_{cx}) \quad (2)$$

where c_j, x_i are the concentrations (moles per volume of solution) of components $\hat{c}_j$ and complexes $\hat{x}_i$, and K_i is the equilibrium constant for the ith reaction. The statement of conservation of mass for each component is

$$u_k = c_k + \sum_{i=1}^{N_{cx}} A_{ik}x_i \ (k = 1, \ldots, N_c), \quad (3)$$

where u_k is the prescribed total soluble concentration of component k. Substituting Equations (2) into (3) yields

$$u_k = c_k + \sum_{i=1}^{N_{cx}} A_{ik}K_i \prod_{j=1}^{N_c} c_j^{A_{ij}} \ (k = 1, \ldots, N_c). \quad (4)$$

Equation (4) represents N_c equations for the N_c unknowns c_k, $k = 1, 2, \ldots, N_c$.

The solution of (4) by a Newton–Raphson iteration can be detailed easily by introducing a vector notation as follows. Let $\mathbf{u}$ be the vector with components u_k, $\mathbf{c}$ the vector with components c_k and $\mathbf{f(c)}$ the vector with components

$$f_k = \sum_{i=1}^{N_{cx}} A_{ik}K_i \prod_{j=1}^{N_c} c_j^{A_{ij}} \ (k = 1, \ldots, N_c). \quad (5)$$

Now define the vector $\mathbf{g(c)}$ as

$$\mathbf{g(c)} = \mathbf{c} + \mathbf{f(c)} - \mathbf{u}. \quad (6)$$

Then Equation (4) is equivalent to

$$\mathbf{g(c)} = \mathbf{0}. \quad (7)$$

Let the $m+1$st iterate of $\mathbf{c}$ be related to the mth iterate by

$$\mathbf{c}^{m+1} = \mathbf{c}^m + \Delta\mathbf{c}^m, \quad (8)$$

and evaluate $\mathbf{f}(\mathbf{c}^{m+1})$ by using a first-order Taylor series

$$\mathbf{f}(\mathbf{c}^{m+1}) = \mathbf{f}(\mathbf{c}^m) + (D\mathbf{f}(\mathbf{c}^m))\Delta\mathbf{c}^m, \quad (9)$$

where $D\mathbf{f(c)}$ is the Jacobian matrix with components $\partial f_k/\partial c_j$. Now evaluate (7) at the $m+1$st iterate

$$\mathbf{g}(\mathbf{c}^{m+1}) = \mathbf{c}^m + \Delta\mathbf{c} + \mathbf{f}(\mathbf{c}^m) + (D\mathbf{f}(\mathbf{c}^m))\Delta\mathbf{c}^m - \mathbf{u} = \mathbf{0}. \quad (10)$$

Thus

$$(\mathbf{I} + D\mathbf{f}(\mathbf{c}^m))\Delta\mathbf{c}^m = -(\mathbf{c}^m + \mathbf{f}(\mathbf{c}^m) - \mathbf{u}) = -\mathbf{g}(\mathbf{c}^m), \quad (11)$$

and

$$\Delta\mathbf{c}^m = -(\mathbf{I} + D\mathbf{f}(\mathbf{c}^m))^{-1}\mathbf{g}(\mathbf{c}^m), \quad (12)$$

where $\mathbf{I}$ is the identity matrix. Equation (12) is solved repeatedly with $\mathbf{c}^{m+1}$ updated according to (8). Iteration is stopped when $\|\mathbf{g}^m\|$ is less than some prescribed tolerance, where $\|\cdot\|$ is any finite dimensional vector norm. The Newton–Raphson scheme (12) possesses second order convergence (Ortega and Rheinboldt, 1970). After the component concentrations are determined the complex concentrations are computed from (2).

CHEMICAL EQUILIBRIUM WITH A SOLID PHASE

A solid species, like the complexes, can be considered the product in a reaction where the components are the reactants

$$\hat{p}_i \rightleftharpoons \sum_{k=1}^{N_c} B_{ik}\hat{c}_k \ (i = 1, \ldots N_s) \quad (13)$$

where $\hat{p}_i$ represents the chemical formula for solid i and the B_{ik} are stoichiometric coefficients. The equilibrium statement for (13) is,

$$K_i^{SO} = \prod_{k=1}^{N_c} c_k^{B_{ik}}, \tag{14}$$

where K_i^{SO} is the solubility product. The concentration of the precipitate, p_i, does not occur in (14) because the activity of the solid is one.

The revised mass balance, to account for the solids, is written

$$h_k \equiv g_k + \sum_{l=1}^{N_s} B_{lk} p_l = 0 \; (k = 1, \ldots, N_c) \tag{15}$$

where g_k are the components of the vector **g** introduced in Equation (7) and h_k is introduced to represent the full set of equations and to simplify subsequent expressions. Equation (15) represents N_c equations for the $N_c + N_s$ unknowns c_i, $i = 1, 2, \ldots, N_c$ and p_i, $i = 1, 2, \ldots, N_s$. The remaining N_s equations are the mass action Equation (14) rewritten as

$$q_i \equiv \prod_{k=1}^{N_c} c_k^{B_{ik}} / K_i^{SO} - 1 = 0 \; (i = 1, \ldots, N_s). \tag{16}$$

Again, the q_i are introduced to consolidate some of the mathematics to follow. It should be noted that Equation (16) can be written more properly as an inequality; less than or equal to zero. If a set of equilibrium concentrations can be obtained and q_i is less than zero, then the solution is undersaturated and the ith reaction in Equation (13) does not take place. If the equilibrium concentrations for the homogeneous system are such that q_i is greater than zero, then the ith reaction in Equation (13) will take place so that q_i is zero. In this sense Equation (16) can be considered a set of inequality constraints on the system. In the development to follow, it is presumed that the reactions in Equation (13) are occurring.

Let **q(c)** be the vector with components defined in (16) and let **B** be the $N_s \times N_c$ matrix with components the stoichiometric coefficients, B_{ik}, and let **h** and **p** be the vectors with components defined in (15). Then (15) and (16) may be rewritten compactly as

$$\mathbf{h(c, p)} = \mathbf{c} + \mathbf{f(c)} + \mathbf{B}^T\mathbf{p} - \mathbf{u} = \mathbf{0} \tag{17}$$

$$\mathbf{q(c)} = \mathbf{0}. \tag{18}$$

Thus, according to (17) and (18), the formation of the solid species defined by the reaction (13) expands the size of the problem fron N_c unknowns to $N_c + N_s$ unknowns. Some researchers (e.g. Reed, 1982) proceed by solving (17) and (18) by a Newton–Raphson iteration. To reduce the size of the system, however, some take advantage of the fact that **h** is linear in **p**. Thus by linear combinations of the equations in (17) the precipitate concentrations may be eliminated reducing (17) to $N_c - N_s$ independent equations. Then along with (18) there are N_c total equations for the components of **c**. Once **c** is determined a subset of the equations in (17) may be used to solve for **p**. This last step involves solving N_s *linear* equations. Thus this approach replaces the problem of $N_c + N_s$ nonlinear equations with N_c nonlinear equations and N_s linear equations. This method has been used recently by Walsh (1983). The elimination of **p** as a primary unknown will be demonstrated here for the simple case of a single precipate. For $N_s = 1$ write (15) as

$$h_k = c_k + f_k + B_{1k} p_1 - u_k = 0. \tag{19}$$

Assume B_{1l} is nonzero, where l is some integer between 1 and N_s. Then

$$h_k - \left(\frac{B_{1k}}{B_{1l}}\right) h_l = c_k + f_k - u_k - \left(\frac{B_{1k}}{B_{1l}}\right) \times (c_l + f_l - u_l) = 0 (k = 1, \ldots, N_c). \quad (k \neq l) \tag{20}$$

Equation (20) represent $N_c - 1$ equations; the remaining equation is from (16)

$$q_1 = \prod_{k=1}^{N_c} c_k^{B_{1k}} / K_1^{SO} - 1 = 0. \tag{21}$$

Once (20) and (21) are solved for the component concentrations c_k, Equation (19) easily yields p_1.

An alternative approach reduces the size of the primary system even further. Rewrite (21) as

$$q_1 = (c_l^{B_{1l}}) \left(\prod_{\substack{k=1 \\ k \neq l}}^{N_c} c_k^{B_{1k}} \right) \Big/ K_1^{SO} - 1 = 0. \tag{22}$$

Now solve (22) for c_l

$$c_l = \left\{ K_1^{SO} \Big/ \prod_{\substack{k=1 \\ k \neq l}}^{N_c} c_k^{B_{1k}} \right\}^{(1/B_{1l})}. \tag{23}$$

Substituting (23) into (20) yields $N_c - 1$ equations for the $N_c - 1$ primary unknowns c_k, $k = 1, \ldots, N_c$ $(k \neq l)$. This last approach is utilized by Morel and Morgan (1972). In general they reduce the $N_c + N_s$ original equations to $N_c - N_s$ primary unknowns plus N_s secondary components determined from the primary set by equations of the form of (23) and finally the precipitates are determined by solving the linear set derived from (17). A drawback to this approach arises because Equation (16) really represents inequalities not equations, that is if $q_i < 0$ then the corresponding reaction in (13) does not take place. Only when the solution is saturated, $q_i = 0$, is p_i formed. Thus, a priori it is not known normally which p_i should be included in (17) with the corresponding constraint equation activated. The usual approach is to solve the system assuming no precipitates are present and then check all the solubility "constraints". If any of the constraints are violated, assume one solid is formed and resolve. Continue in this manner until a converged solution satisfies all solubility constraints. Thus the size of the problem is continually

changing and the expressions for the terms in the coefficient matrix must be recalculated after every addition or deletion of a solid.

To recap, three basic approaches have been taken for handling the presence of precipitates. In the first (Reed, 1982, for example) the addition of N_s solids increases the size of the system to $N_c + N_s$. After some manipulations, Walsh (1983) converts the size of the system to N_c; the original size without solids. Finally Morel and Morgan (1972), again after algebraic manipulations reduce the system to $N_c - N_s$ equations for the primary components. In all situations the resulting system of algebraic equations are solved by a Newton–Raphson iteration. It seems that a direct comparison between these approaches has not yet been made. Although solving a smaller set seems the most attractive, the accompanying algebraic manipulations must be considered as well as the conditioning of the resulting matrices.

In this paper a method is presented which similar to Walsh retains the system size as N_c. However no transformation of the equation is required; an auxiliary matrix, which represents the contribution of the solids, is added to the coefficient matrix. An attractive feature is the ease with which precipitation–dissolution reactions are included.

PENALTY METHOD

Equations (17) and (18) represent $N_c + N_s$ equations for the unknowns, **c** and **p**. A distinctive feature of these equations is that they are not coupled in all the variables, that is the unknowns in the vector **p** do not occur in the constraint Equation (18). The components of **p** are similar to Lagrange multipliers in a constrained minimization problem and may be eliminated as primary unknowns by a penalty function method similar to that employed in classical optimization. However a better motivation for the penalty function method is to consider the equilibrium concentration of the solid to be the limit of an expression obtained from the kinetic rate law for the dissolution reaction. This limit can be approached via some scalar parameter, not necessarily real time. This motivational argument for the penalty function approach is detailed in Appendix 1. Following this argument Equations (17) and (18) are replaced with

$$\mathbf{h}_\alpha(\mathbf{c}_\alpha) = \mathbf{g}(\mathbf{c}_\alpha) + \alpha \mathbf{B}^T \mathbf{q}(\mathbf{c}_\alpha) = 0, \tag{24}$$

where α is termed the penalty parameter. Equation (24) is simply (17) with **p** replaced by α **q**. The designation $\mathbf{c}_\alpha$ indicates that the solution vector depends on α. The following limits follow under suitable restrictions,

$$\lim_{\alpha \to \infty} \mathbf{c}_\alpha = \mathbf{c} \text{ [solution of (17)(18)]} \tag{25}$$

$$\lim_{\alpha \to \infty} \alpha \mathbf{q}(\mathbf{c}_\alpha) = \mathbf{p}. \tag{26}$$

It has been determined that sufficiently accurate answers are obtained by selecting a single large value of α. If double precision calculations are used, the range of values of α which yield accurate answers is wide. This will be discussed further in the next section. From hereon the subscript α will be dropped but the dependence of the solution on α should be kept in mind.

It is instructive to examine the Taylor series expansion of **h** which yields the Newton–Raphson iteration

$$(\mathbf{I} + D\mathbf{f}(\mathbf{c}^m) + \alpha \mathbf{B}^T D\mathbf{q}(\mathbf{c}^m))\Delta \mathbf{c}^m = -\mathbf{h}(\mathbf{c}^m). \tag{27}$$

The difference between (27) and (11) (the iteration scheme for the system without solids) is simply the addition of the matrix $\alpha \mathbf{B}^T D\mathbf{q}(\mathbf{c}^m)$ to the coefficient matrix. Therefore programs which solve only homogeneous systems can be modified easily to account for precipitation–dissolution reactions. This is the primary purpose and the major advantage of the penalty approach.

The assembly of the additional matrix is a simple matter because,

$$\frac{\hat{c} q_k}{\hat{c} c_j} = \frac{(q_k + 1) B_{kj}}{c_j}. \tag{28}$$

Thus

$$(\mathbf{B}^T D\mathbf{q}(\mathbf{c}))_{ij} = \frac{1}{c_j} \sum_{k=1}^{N_s} B_{ki} B_{kj} (q_k + 1). \tag{29}$$

For computation purposes it must be noted that each q_i is actually an inequality constraint on the system and therefore if $q_i(\mathbf{c}^m)$ is less than zero, that constraint is not included in the next iteration. That is, a constraint is only activated once it is violated. When there is more than one solid possible in a system, various approaches for activating the constraints are possible. For instance, Walsh (1983) and Morel and Morgan (1972) activate one constraint, completely solve the equilibrium problem and then check solubility products to see if any constraints are violated; if so, activate another constraint and reequilibrate. In the sample problems presented here, only one solid is present and the equilibrium solution ignoring the precipitation reaction violates the solubility constraint and thus it must be included as an equality constraint.

EXAMPLE CALCULATIONS

First, to demonstrate explicitly the calculations, consider the simple system governed by the following reactions

$$\hat{c}_1 + \hat{c}_2 \rightleftharpoons \hat{x}_1;\ K_1 \tag{30}$$

$$\hat{p}_1 \rightleftharpoons \hat{c}_2 + \hat{c}_3;\ K_1^{SO}. \tag{31}$$

The mass balances for the three components can be written, after substituting the mass action equations, as

$$\begin{aligned} h_1 &= c_1 + K_1 c_1 c_2 - u_1 = 0, \\ h_2 &= c_2 + K_1 c_1 c_2 - u_2 = 0, \\ h_3 &= c_3 + p - u_3 = 0, \end{aligned} \tag{32}$$

and the solubility constraint is

$$q_1 = c_2c_3/K_1^{SO} - 1 = 0. \quad (33)$$

Equations (32) and (33) are a specific example of (17) and (18). The penalty function formulation for this system corresponding to Equation (24) is

$$\begin{aligned} h_1(\mathbf{c}) &= c_1 + f_1(\mathbf{c}) + \alpha B_1 q(\mathbf{c}) - u_1 = 0 \\ &= c_1 + K_1c_1c_2 - u_1 = 0 \\ h_2(\mathbf{c}) &= c_2 + f_2(\mathbf{c}) + \alpha B_2 q(\mathbf{c}) - u_2 = 0 \\ &= c_2 + K_1c_1c_2 + \alpha(1) \\ &\quad \times (c_2c_3/K_1^{SO} - 1) - u_2 = 0 \\ h_3(\mathbf{c}) &= c_3 + f_3(\mathbf{c}) + \alpha B_3 q(\mathbf{c}) - u_3 = 0 \\ &= c_3 + \alpha(1)(c_2c_3/K_1^{SO} - 1) - u_3 = 0, \end{aligned} \quad (34)$$

where the fact that B_1 and f_3 are zero has been used. The matrices required for Equation (27) are (for the special situation of one solid, $\mathbf{B}$ and $D\mathbf{q}$ are vectors)

$$D\mathbf{f}(\mathbf{c}) = \begin{bmatrix} \partial f_1/\partial c_1 & \partial f_1/\partial c_2 & \partial f_1/\partial c_3 \\ \partial f_2/\partial c_1 & \partial f_2/\partial c_2 & \partial f_2/\partial c_3 \\ \partial f_3/\partial c_1 & \partial f_3/\partial c_2 & \partial f_3/\partial c_3 \end{bmatrix},$$

$$= \begin{bmatrix} K_1c_2 & K_1c_1 & 0 \\ K_1c_2 & K_1c_1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad (35)$$

and

$$\mathbf{B}^T = \begin{Bmatrix} 0 \\ 1 \\ 1 \end{Bmatrix}, \quad (36)$$

$$D_q(\mathbf{c}) = (0, c_3, c_2)/K^{SO}. \quad (37)$$

Thus the Newton–Raphson iteration scheme for this sample problem corresponding to Equation (27) is

$$\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} K_1c_2^m & K_1c_1^m & 0 \\ K_1c_2^m & K_1c_1^m & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{\alpha}{K_1^{SO}} \begin{Bmatrix} 0 \\ 1 \\ 1 \end{Bmatrix} (0 \;\; c_3^m \;\; c_2^m) \right) \begin{Bmatrix} \Delta c_2^m \\ \Delta c_2^m \\ c_3^m \\ 3 \end{Bmatrix} = \begin{Bmatrix} h_1(\mathbf{c}^m) \\ h_2(\mathbf{c}^m) \\ h_3(\mathbf{c}^m) \end{Bmatrix}. \quad (38)$$

Using the data from Table 1, with $\alpha = 1000$, and inital conditions, $c_1^0 = 0.7402$, $c_2^0 = 3.1403$, $c_3^0 = 4.4$, yields the following system of equations for the first iteration.

$$\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 3.512 & 0.5922 & 0 \\ 2.512 & 0.5922 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 733.33 & 523.379 \\ 0 & 733.33 & 523.379 \end{bmatrix} \right) \begin{Bmatrix} \Delta c^0 1 \\ \Delta c^0 2 \\ \Delta c^0 3 \end{Bmatrix} = - \begin{Bmatrix} 0.000006 \\ 1302.866 \\ 1302.866 \end{Bmatrix}.$$

Solution of this set yield $\Delta\mathbf{c}^0 = (0.163196, -0.96786, -1.13105)$ and therefore $\mathbf{c}^1 = (0.90347, 2.1724, 3.26894)$. This procedure then is repeated until the residual vector $\mathbf{h}(\mathbf{c}^m)$ becomes sufficiently small. The final converged values for $\mathbf{c}$ and $\mathbf{p}$ are given in Table 1. Notice that accurate answers are obtained for α ranging through many orders of magnitude. In each situation only three iterations were required, using double precision. The convergence criteria used was $\|\mathbf{h}^m\|_2 < 10^{-5}\|\mathbf{h}^0\|_2$; where $\|\mathbf{h}^m\|_2$ is the square root of the sum of the squares of the components of $\mathbf{h}$ at the mth iterate. For $\alpha > 10^{13}$ the accuracy deteriorates due to ill-conditioning of the matrix.

The second example, although rather simple, is more realistic. The problem is to determine the equilibrium composition of a solution to which 10^{-3} moles/l of $CaCO_3$ is added. This example is taken from Appendix 2 of Westall, Zachary, and Morel (1976). The system consists of three components, six

Table 1. Results for example problem No. 1

Approximate Solutions				
Log α	c_1	c_2	c_3	p
0	.943	2.196	3.254	1.146
1	.9926	2.024	3.032	1.368
2	.9992	2.002	3.003	1.397
3	.9999	2.000	3.000	1.399
4	.9999	2.000	3.000	1.399
5	1.000	2.000	3.000	1.400

Data: $K_1 = 0.80$, $K^{SO} = 6.00$, $u_1 = 2.6$, $u_2 = 5.0$, $u_3 = 4.4$.
Exact solution: $c_1 = 1.0$, $c_2 = 2.0$, $c_3 = 3.0$, $p = 1.4$.
Initial guess: $c_1^0 = 0.7402$, $c_2^0 = 3.1403$, $c_3^0 = 4.40$, from solution assuming no solid.

Table 2. Data for calcium carbonate sample problem

	$\hat{c}_1 = Ca^{2+}$	$\hat{c}_2 = H^+$	$\hat{c}_3 = CO_3^{2-}$	$\log K_i$
$\hat{x}_1 = CaCO_3(aq)$	1	0	1	3.0
$\hat{x}_2 = CaHCO_3^+$	1	1	1	11.6
$\hat{x}_3 = CaOH^+$	1	−1	0	−12.2
$\hat{x}_4 = HCO_3^-$	0	1	1	10.2
$\hat{x}_5 = H_2CO_3$	0	2	1	16.5
$\hat{x}_6 = OH^-$	0	−1	0	−14.0

A matrix

	$\hat{c}_1$	$\hat{c}_2$	$\hat{c}_3$	$\log \hat{K}_i^0$
$\hat{p}_1 = CaCo_3(s)$	1	0	1	8.3

B matrix

Prescribed concentrations: $u_1 = 10^{-3}$M/l, $u_2 = 0$, $u_3 = 10^{-3}$M/l.
N.B.: $Ca(OH)_2(s)$ is not considered in this sample problem. Its solubility constraint is not violated, however, by the final solution.

aqueous complexes, and one solid. The data, that is the stoichiometric coefficients A_{ij} and B_{ij}, the equilibrium constants, and the total component concentration in moles/l of each component are given in Table 2. The solution with and without the solubility constraint (expressed as $-\log$ concentration) obtained by the penalty function method described herein is given in Table 3. It should be noted that the solution required six iterations (double precision calculations) when using as initial guess for the component concentrations the equilibrium solution assuming no solid present. The convergence criteria used was the same as in the first problem. Of course convergence of $\|\mathbf{h}^m\|_2$ does not imply $q_1 \sim 0$. However for $10^2 < \alpha < 10^8$, q_1 always is $< 10^{-5}$. Thus the method works, for this problem, through a wide range of penalty parameters. Details of the calculation for the first iteration are provided in Appendix 2.

COMPUTER PROGRAM

Appendix 3 contains the FORTRAN program PENALTY, which will perform the calculations described in this paper. The program has been run using Microsoft FORTRAN on a Macintosh Plus. All documentation required is in the program. Variable definitions in the program follow closely the notation in the paper. Following the program are the input and output for the two sample problems discussed here.

DISCUSSION

The penalty function method presented here is an alternative method for incorporating precipitation-dissolution reactions in the equilibrium constant method for solving the chemical equilibrium problem. From a programming viewpoint the method is simple to incorporate into existing codes which do not hand-

Table 3. Results for calcium carbonate sample problem, with and without solid

Species	no solid $-\log[x_i]$	with $CaCO_3(s)$ $-\log[x_i]$
Ca^{2+}	3.163	3.915
H^+	10.413	9.9066
CO_3^{2-}	3.364	4.384
$CaCO_3$	3.526	5.299
$CaHCO_3^+$	5.339	6.6066
$CaOH^+$	4.950	6.208
HCO_3^-	3.576	4.091
H_2CO_3	7.689	7.698
OH^-	3.587	4.093
$CaCO_3(s)$	–	3.059

Answers obtained with $\alpha = 100$. Solution of $-\log$ concentration is accurate to 4 significant figures for $10^2 < \alpha < 10^8$.

le precipitation–dissolution reactions. Also no variable eliminations need to be performed and the size of the system always is the same; even as additional solids enter the problem. The efficiency of this technique relative to existing methods can be studied only with particular applications in mind, because the conditioning of the matrices surely plays a role.

It also should be mentioned that there are extensions of this method worthy of examination. For instance the so-called augmented Lagrangian methods (Fortin and Glowinski, 1982) may prove effective.

A study is currently underway to compare the various approaches which have been discussed herein.

Acknowledgment—The work presented in this paper was supported by grant DE-AC02-79EV10253, from the Ecological Research Division, Office of Energy Research, U.S. Department of Energy to the University of Notre Dame.

REFERENCES

Brinkley, S. R., 1946, Note on the conditions of equilibrium for systems of many constituents: Jour. Chem. Phys., v. 14, p. 563–564.

Brinkley, S. R., 1947, Calculation of the equilibrium composition of systems of many constituents: Jour. Chem. Phys., v. 15, p. 107–110.

Fortin, M., and Glowinski, R., 1982, Augmented Lagrangian methods: applications to the numerical solution of boundary value problems: Elsevier Science Publ., New York, 1982, 340 p.

Kirkner, D. J., Theis, T. L., and Jennings, A. A., 1984, Multicomponent solute transport with sorption and soluble complexation: Advances in Water Resources, v. 7, p. 120–125.

Morel, F., and Morgan, J., 1972, A numerical method for computing equilibria in aqueous chemical systems: Environmental Science and Technology, v. 6, p. 58–67.

Morin, K. A., 1985, Simplified explanations and examples of computerized methods for calculating chemical equilibrium in water: Computers & Geosciences, v. 11, no. 4, p. 409–416.

Nordstrom, D. K., Plummer, L. N., Wigley, T. M. L., Wolery, T. J., Ball, J. W., Jenne, E. A., Bassett, R. L., Crerar, D. A., Florence, T. B., Fritz, B., Hoffman, M., Holdren, G. R., Jr., Lafon, G. M., Mattigod, S. V., McDuff, R. E., Morel, F., Reddy, M. M., Sposito, G., and Thrailkill, J., 1979, A comparison of computerized chemical models for equilibrium calculations in aqueous systems, *in* Jenne, E. A., ed., Chemical modeling in aqueous systems: Am. Chemical Society, Washington, D.C., p. 857–892.

Ortega, J. M., and Rheinboldt, W. C., 1970, Iterative solution of nonlinear equations in several variables: Academic Press, New York, 502 p.

Reed, M. H., 1982, Calculation of multicomponent chemical equilibria and reaction processes in systems involving minerals, gases and an aqueous phase: Geochimica et Cosmochimica Acta, v. 46, p. 513–528.

Walsh, M. P., 1983, Geochemical flow modeling: unpubl. doctoral dissertation, Univ. Texas-Austin, 502 p.

Walsh, M. P., Bryant, S. L., Schechter, R. S., and Lake, L. W., 1984, Precipitation and dissolution of solids attending flow through porous media: AICHE Jour., v. 30, p. 317–328.

Westall, J. C., Zachary, J. L., and Morel, F. M. M., 1976, MINEQL—A computer program for the calculation of chemical equilibrium composition of aqueous systems: Tech. Note No. 18, Massachusetts Inst. of Technology, Dept. of Civil Engineering, 91 p.

NOMENCLATURE

A_{ij} = Stoichiometric coefficient for aqueous phase complexation reaction
B_{ij} = Stoichiometric coefficient for precipitation reactions
$\hat{c}_j$ = Chemical formula of component j
c_j = Concentration of component j (moles per unit volume of solution)
k_b = Backward reaction rate constant
k_f = Forward reaction rate constant
K_i = Equilibrium constant for the ith aqueous phase complexation reaction
K_i^{so} = Solubility product governing ith precipitation reaction
N = Total number of constituents in the aqueous system
N_c = Number of components
N_{cx} = Number of complexes
N_s = Number of precipitated solids
$\hat{p}_i$ = Chemical formula of precipitate i
p_i = Concentration of precipitate i (moles per unit volume of solution)
$q_i(\mathbf{c})$ = Constraint equation for ith precipitation reaction
$\hat{x}_i$ = Chemical formula of complex i
x_i = Concentration of complex i (moles per unit volume of solution)
α = Penalty parameter
Δ = Discrete increment in a variable

Matrix and vector quantities

$\mathbf{A}$ = Stoichiometric matrix for aqueous phase complexation reactions
$\mathbf{B}$ = Stoichiometric matrix for precipitation reactions
$\mathbf{Df(c)}$ = Jacobian matrix with members $\partial f_i/\partial c_j$
$\mathbf{I}$ = Identity matrix
$\mathbf{c}$ = Vector of component concentrations
$\mathbf{f(c)}$ = Vector valued function of component vector $\mathbf{c}$
$\mathbf{g(c)}$ = Vector of conservation of mass expressions for each component
$\mathbf{h(c)}$ = Vector of revised conservation of mass expressions to account for precipitation reactions
$\mathbf{p}$ = Vector of precipitation concentrations
$\mathbf{q(c)}$ = Vector of constraint equations for precipitation reactions
$\mathbf{u}$ = Vector of total soluble component concentrations

APPENDIX 1

The penalty function approach can be motivated by the following arguments. Consider, for example, the chemical kinetic rate expressions for the reaction of Equation (31),

$$\frac{dp_1}{dt} = k_b c_2 c_3 - k_f, \tag{A1}$$

where t is time, k_f and k_b are the forward and backward rate constants respectively and the activity of the solid is taken as one. We assume order corresponds to stoichiometry. Equation (A1) also can be written

$$\frac{dp_1}{dt} = k_f(c_2c_3/K^{SO} - 1), \tag{A2}$$

where $K^{SO} = k_f/k_b$ is the equilibrium constant for the reaction. We define the expression in the parentheses to be q_1 and rewrite (A2) as

$$\frac{dp_1}{dt} = k_f q_1(t). \tag{A3}$$

We know that at equilibrium dp_1/dt is zero and therefore $q_1(t)$ is zero at equilibrium also. Additionally, because the concentrations must be positive, the function $q_1(t)$ always is $\geqslant -1$ and also is bounded above because dp_1/dt must be finite. With these properties of q_1 in mind, we now integrate Equation (A3) yielding

$$p_1(T) = k_f \int_0^T q_1(t)dt, \tag{A4}$$

assuming no solid is initially present. T is some specific time. Using the mean value theorem, Equation (A5) can be written

$$p_1(T) = k_f T \overline{q_1(T)}, \tag{A5}$$

where $\overline{q_1(T)}$ is the mean value of q_1 over the interval $(0, T)$. Taking the limit of both sides of Equation (A5) as T goes to infinity and noting that $p_1(T)$ is bounded, implies

$$\lim_{T \to \infty} \overline{q_1(T)} = 0. \tag{A6}$$

But from the properties of $q_1(t)$ as discussed,

$$\lim_{T \to \infty} \overline{q_1(T)} = q_1(\infty) = 0. \tag{A7}$$

Thus as T goes to infinity, $k_f T\overline{q_1(T)}$ has a limit which is the equilibrium value of p_1, and $\overline{q_1(T)}$ approaches the equilibrium value of $q_1(t)$. Because T only needs to be considered as a parameter and not as real time, the equilibrium value of p_1 can be written as

$$p_1 = \lim_{\alpha \to \infty} \alpha q_1(\alpha), \tag{A8}$$

which basically is Equation (26).

Thus, to recap, the equilibrium value of the precipitate concentration may be replaced with the limit of the kinetic rate expression [written as in Eq. (A5)] as some positive, real paramenter goes to infinity.

Although Equation (A8) is simply a consequence of the transient system, normally not exploitable, herein it is determined to be a useful artifice for solving the equilibrium problem.

APPENDIX 2

$CaCO_3$ Example

Components

$\dot{c}_1 = Ca^{2+}$
$\dot{c}_2 = H^+$
$\dot{c}_3 = CO_3^{2-}$

Complexes

$\dot{x}_1 = CaCO_3$ (aq)
$\dot{x}_2 = CaHCO_3$
$\dot{x}_3 = CaOH^+$
$\dot{x}_4 = HCO_3^-$
$\dot{x}_5 = H_2CO_3$
$\dot{x}_6 = OH^-$.

Solid

$\dot{p}_1 = CaCO_3(s)$
$Ca(OH)_2(s)$ not considered.

Reaction: aqueous

$Ca^{2+} + CO_3^{2-} \rightleftharpoons CaCO_3$
$Ca^{2+} + CO_3^{2-} + H^+ \rightleftharpoons CaHCO_3^+$
$Ca^{2+} + OH^- \rightleftharpoons CaOH^+$
$H^+ + CO_3^{2-} \rightleftharpoons HCO_3^-$
$2H^+ + CO_3^{2-} \rightleftharpoons H_2CO_3$

Reactions: solid phase

$$Ca(OH)_2(s) \rightleftharpoons Ca^{2+} - 2H^+$$
$$CaCO_3(s) \rightleftharpoons Ca^{2+} + CO_3^{2-}$$

Imposed concentrations	Initial guesses from solution assuming no precipitation
$u_1 = 10^{-3}$ M/l	$c_1^0 = 6.8707 \times 10^{-4} = 10^{-3.163}$
$u_2 = 0.0$ M/l	$c_2^0 = 3.8637 \times 10^{-11} = 10^{-10.413}$
$u_3 = 10^{-3}$ M/l	$c_3^0 = 4.3251 \times 10^{-4} = 10^{-3.364}$.

This system written in the form of Equation (32) is,

$$h_1(\mathbf{c}) = c_1 + K_1c_1c_3 + K_2c_1c_2c_3 + K_3c_1c_2^{-1} + \alpha\left(\frac{c_1c_3}{K_1^{SO}}\right) - 1 - u_1 = 0.0,$$

$$h_2(\mathbf{c}) = c_2 + K_2c_1c_2c_3 - K_3c_1c_2^{-1} + K_4c_2c_3 + 2K_5c_2^2c_3 - K_6c_2^{-1} - u_2 = 0.0,$$

$$h_3(\mathbf{c}) = c_3 + K_1c_1c_3 + K_2c_1c_2c_3 + K_4c_2c_3 + K_5c_2^2c_3 + \alpha\left(\frac{c_1c_3}{K_1^{SO}} - 1\right) - u_3 = 0.0.$$

The Newton-Raphson form, Equation (33), for this system is,

$$\left(\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} + \begin{bmatrix}(K_1c_3^m + K_2c_2^mc_3^m + K_3(c_2^m)^{-1}) & (K_2^mc_1^mc_3^m - K_2^mc_1^mc_3^m - K_3^m(c_2^m)^{-2})\\ (K_2c_2^mc_3^m - K_3(c_2^m)^{-1}) & (K_2c_1^mc_3^m + K_3c_1^m(c_2^m)^{-2} + K_4c_3^m + 4K_5c_2^mc_3^m + K_6(c_2^m)^{-2})\\ (K_1c_3^m + K_2c_2^mc_3^m) & (K_2c_1^mc_3^m + K_4c_3^m + K_5(c_2^m)^2)\end{bmatrix}\right.$$

$$\left.\begin{matrix}(K_1c_1^m + K_2c_1^mc_2^m)\\(K_2c_1^mc_2^m + K_4c_2^m + 2K_5(c_2^m)^2)\\(K_1c_1^m + K_2c_2^m + K_4c_2^m + K_5(c_2^m)^2)\end{matrix}\right] + \frac{\alpha}{K_1^{SO}}\begin{Bmatrix}1\\0\\1\end{Bmatrix}(c_3 \quad 0 \quad c_1)\Bigg]\begin{Bmatrix}\Delta c_1^m\\\Delta c_2^m\\\Delta c_3^m\end{Bmatrix} = \begin{Bmatrix}h_1(c^m)\\h_2(c^m)\\h_3(c^m)\end{Bmatrix}.$$

The matrices arising from Equation (33) using the data given in Tables 2 and 3 with $\alpha = 100$

$$\left(\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} + \begin{bmatrix}10^{-0.3415} & -10^{5.2358} & 10^{-0.1564}\\ -10^{-2.0132} & 10^{7.145} & 10^{-0.2055}\\ 10^{-0.3573} & 10^{6.8435} & 10^{0.1173}\end{bmatrix} + \begin{bmatrix}10^{6.936} & 0 & 10^{7.137}\\0&0&0\\10^{6.936}&0&10^{7.137}\end{bmatrix}\right)\begin{Bmatrix}\Delta c_1\\\Delta c_2\\\Delta c_3\end{Bmatrix} = -\begin{Bmatrix}10^{3.7656}\\10^{-6.2368}\\10^{3.7656}\end{Bmatrix}.$$

The solution for the first increment is then

$$\Delta c^0 = \begin{Bmatrix}-10^{-3.1854}\\-10^{-13.0572}\\-10^{-5.0054}\end{Bmatrix} \text{ and thus, } c^1 = \begin{Bmatrix}10^{-4.462}\\10^{-10.414}\\10^{3.374}\end{Bmatrix}.$$

APPENDIX 3

Program PENALTY

```
C       PROGRAM PENALTY
C.....PROGRAM TO CALCULATE BATCH EQUILIBRIUM FOR Nc COMPONENTS
C.....AND Ncx COMPLEXES.  THE PROGRAM USES THE PENALTY FUNCTION
C.....METHOD TO ENFORCE THE SOLUBILITY CONSTRAINT FOR Ns SOLIDS
```

```
C.....FORMED BY THE PRECIPITATION OF COMPONENTS.   ALL DATA IS
C.....INPUT BY THE USER, EITHER FROM THE KEYBOARD OR DATA FILES,
C.....DATA FILES ARE CREATED IF THE KEYBOARD IS USED TO ENTER
C.....DATA INITIALLY.
C.....THE PENALTY FUNCTION PROCEDURE MAY BE USED IN OTHER BATCH
C.....EQUILIBRIUM PROGRAMS IN A MANNER SIMILAR TO THAT EMPLOYED
C.....HERE.
C.....VARIABLE LIST:
C         A(I,J) = STOICHIOMETRIC MATRIX FOR COMPLEXES,(INTEGER),
C         ALPHA = PENALTY PARAMETER
C         B(I,J) = STOICHIOMETRIC MATRIX FOR PRECIPITATES,(INTEGER),
C         C(I) = CONCENTRATION OF COMPONENT I,
C         CMPLX(I) = NAME OF COMPLEX I,(CHARACTER*8),
C         COMP(I) = NAME OF COMPONENT I,(CHARACTER*8),
C         DELTA = CONVERGENCE CRITERIA, NORM(DC)/NORM(DC0)<DELTA,
C         DF(I,J) = JACOBIAN MATRIX FOR COMPLEXES, DF(I,J)=dF(I)/dC(J)
C         DQ(I,J) = JACOBIAN MATRIX FOR PRECIPITATES * B(TRANSPOSE),
C         F(I) = FUNCTION REPRESENTING MASS OF COMPONENT I
C                IN COMPLEXED FORMS,
C         G(I) = RIGHT HAND SIDE OF MATRIX EQUATION
C                G(I) = -(C(I)+F(I)-U(I) + PENALTY TERMS IF NEEDED),
C         IBATCH = FLAG VARIABLE,
C                  IF 0  - SOLVE EQUIBRIUM PROBLEM WITHOUT SOLID
C                          AND THEN CHECK SOLUBILITY CONSTRAINT
C                  IF NOT 0-CHECK SOLUBILITY CONSTRAINT FOR INITIAL
C                          GUESS AND EACH ITERATION THEREAFTER,
C         ITER = COUNTER FOR NUMBER OF ITERATIONS,
C         ITERMAX = MAXIMUM NUMBER OF ITERATIONS ALLOWED,
C         ITABLE = COUNTER USED IN OUTPUT TO STORE NUMBER OF
C                  STOICHIOMETRIC MATRIX TABLES REQUIRED,
C         JCOMP = INTEGER READ IN TO GIVE COMPONENT NUMBER IN COMPLEX,
C         JSTOICH = STOICHIOMETRIC COEFFICIENT OF COMPONENT JCOMP IN
C                   THE CURRENT COMPLEX,
C         NC = NUMBER OF COMPONENTS,
C         NCX = NUMBER OF COMPLEXES,
C         NCOMP(I) = NUMBER OF COMPONENTS IN COMPLEX I,
C         NPPT(I) = NUMBER OF COMPONENTS IN PRECIPITATE I,
C         NS = NUMBER OF SOLIDS,
C         P = "CONCENTRATION" OF PRECIPITATE,
C         PRECIP = LOGICAL VARIABLE, USED TO INDICATE IF THE
C                  SOLUBILITY CONSTRAINT IS VIOLATED,
C         PPT(I) = NAME OF PRECIPITATE I,(CHARACTER*8),
C         PROD = VARIABLE USED TO STORE AND CALCULATE PRODUCTS,
C         Q(I) = SOLUBILITY CONSTRAINT FOR SOLID I,
C         RK(I) = EQUILIBRIUM COEFFICIENT FOR COMPLEX I,
C         RKSO(I) = SOLUBILITY PRODUCT FOR PRECIPITATE I,
C         RLHS(I,J) = LEFT HAND SIDE MATRIX,
C         U(I) = TOTAL PRESCRIBED (ANALYTICAL) CONCENTRATION OF
C                COMPONENT I,
C         X = CONCENTRATION OF COMPLEX.
C
C     REQUIRED SUBROUTINES:
C         MATRIX SOLVER, FOR THIS PROGRAM THE SOLVER GAUSS()
C         IS USED.
C
C     REQUIRED FILES:
C         8 = FILE TO STORE INPUT DATA.  THIS FILE IS CREATED IF
C             THE KEYBOARD IS USED TO INPUT DATA OR IS THE FILE USED
C             IF INPUT FROM A FILE IS SELECTED.  THE NAME OF THE
C             FILE IS INPUT BY THE USER AND IS STORED IN THE
C             VARIABLE INDATA, (CHARACTER*15)
C        10 = FILE TO WRITE OUTPUT DATA. THE FILE NAME IS ENTERED BY
C             THE USER AND STORED IN THE VARIABLE OUTPUT, (CHAR*15)
C
      IMPLICIT REAL*8(A-H,O-Z)
      CHARACTER*8 COMP(10),CMPLX(10),PPT(10)
      CHARACTER*15 INDATA,OUTPUT
      INTEGER A(10,10),B(10,10)
      LOGICAL PRECIP,BATCH
```

```
      DIMENSION C(10),F(10),G(10),TITLE(20)
      DIMENSION RK(10),RKSO(10),NCOMP(10),NPPT(10),U(10)
      DIMENSION DF(10,10),Q(10),DQ(10,10),RLHS(10,10)
C
C.....READ IN DATA - UNIT 9 IS TERMINAL FOR MICROSOFT FORTRAN
C.....GET OUTPUT FILE NAME AND OPEN FILE AS NEW
C.....EXISTING FILES WITH THE SAME NAME ARE FIRST DELETED
      WRITE(9,*)' ENTER THE NAME OF A FILE TO WRITE OUTPUT'
      READ(9,*)OUTPUT
      OPEN(10,FILE=OUTPUT,STATUS='NEW')
C.....DETERMINE INPUT TYPE, FROM FILE OR KEYBOARD
      WRITE(9,*)' TYPE "0" TO ENTER DATA FROM A FILE'
      WRITE(9,*)' OR "1" TO ENTER DATA FROM THE KEYBOARD'
      READ(9,*)DATAIN
      IF(DATAIN .EQ. 0)GO TO 50
C.....DATA INPUT FROM THE KEYBOARD
C.....ENTER TITLE
      WRITE(9,*)' ENTER TITLE (ONE LINE)'
      READ(9,1000)TITLE
C.....ENTER PENALTY PARAMETER AND CONVERGENCE CRITERIA
      WRITE(9,*)' ENTER PENALTY PARAMETER, ALPHA'
      WRITE(9,*)' AND CONVERGENCE CRITERIA'
      READ(9,*)ALPHA,DELTA
C.....ENTER NUMBER OF COMPONENTS, COMPLEXES AND PRECIPITATES
      WRITE(9,*)' ENTER THE NUMBER OF COMPONENTS, Nc'
      WRITE(9,*)' THE NUMBER OF COMPLEXES, Ncx'
      WRITE(9,*)' AND THE NUMBER OF PRECIPITATED SOLIDS, Ns'
      READ(9,*)NC,NCX,NS
C.....GET THE NAMES OF THE COMPONENTS, COMPLEXES AND SOLIDS
      DO 10 I=1,NC
      WRITE(9,*)' ENTER THE NAME OF COMPONENT',I
   10 READ(9,*)COMP(I)
      DO 11 I=1,NCX
      WRITE(9,*)' ENTER THE NAME AND EQUILIBRIUM CONSTANT FOR COMPLEX'
     &,I
   11 READ(9,*)CMPLX(I),RK(I)
      DO 12 I=1,NS
      WRITE(9,*)' ENTER THE NAME AND SOLUBILITY PRODUCT FOR PRECIPITATE'
     &,I
   12 READ(9,*)PPT(I),RKSO(I)
C.....GET THE STOICHIOMETRIC MATRICES A AND B.
C.....SET ALL ENTRIES TO ZERO AND HAVE USER ENTER NON-ZERO ENTRIES
      DO 20 I=1,NCX
      DO 20 J=1,NC
   20 A(I,J)=0.0D0
      DO 21 I=1,NS
      DO 21 J=1,NC
   21 B(I,J)=0.0D0
      WRITE(9,*)' YOU MUST NOW ENTER THE STOICHIOMETRIC MATRIX FOR'
      WRITE(9,*)' THE COMPLEXES ENTERED ABOVE'
      WRITE(9,*)' FOR EACH COMPLEX YOU WILL HAVE TO ENTER THE NUMBER'
      WRITE(9,*)' OF COMPONENTS, THEN EACH COMPONENT NUMBER AND'
      WRITE(9,*)' STOICHIOMETRIC COEFFICIENT'
      DO 30 I=1,NCX
C.....GET THE NUMBER OF COMPONENTS IN COMPLEX I
      WRITE(9,*)' ENTER THE NUMBER OF COMPONENTS IN COMPLEX',
     &I,' ',CMPLX(I)
      READ(9,*)NCOMP(I)
      DO 25 J=1,NCOMP(I)
      WRITE(9,*)' ENTER THE NUMBER AND STOICHIOMETRIC COEFFICIENT OF'
      WRITE(9,*)' COMPONENT',J,' OF ',CMPLX(I)
      READ(9,*)JCOMP,JSTOIC
C.....PUT DATA IN MATRIX A
      A(I,JCOMP)=JSTOIC
   25 CONTINUE
   30 CONTINUE
      WRITE(9,*)' YOU MUST NOW ENTER THE STOICHIOMETRIC MATRIX FOR'
      WRITE(9,*)' THE PRECIPITATES ENTERED ABOVE FOR EACH'
      WRITE(9,*)' PRECIPITATE YOU WILL HAVE TO ENTER THE NUMBER'
```

```
      WRITE(9,*)' OF COMPONENTS, THEN EACH COMPONENT NUMBER AND'
      WRITE(9,*)' STOICHIOMETRIC COEFFICIENT'
      DO 40 I=1,NS
C.....GET THE NUMBER OF COMPONENTS IN PRECIPITATE I
      WRITE(9,*)' ENTER THE NUMBER OF COMPONENTS IN PRECIPITATE',
     &I,' ',PPT(I)
      READ(9,*)NPPT(I)
      DO 35 J=1,NPPT(I)
      WRITE(9,*)' ENTER THE NUMBER AND STOICHIOMETRIC COEFFICIENT OF'
      WRITE(9,*)' COMPONENT',J,' OF ',PPT(I)
      READ(9,*)JCOMP,JSTOIC
C.....PUT DATA IN MATRIX B
      B(I,JCOMP)=JSTOIC
   35 CONTINUE
   40 CONTINUE
C.....INPUT TOTAL SOLUBLE COMPONENT CONCENTRATIONS AND INITIAL
C.....GUESSES FOR THE FREE COMPONENT CONCENTRATION AT EQUILIBRIUM
      DO 41 I=1,NC
      WRITE(9,*)' INPUT THE TOTAL SOLUBLE CONCENTRATION AND AN INITIAL
      WRITE(9,*)' GUESS FOR THE CONCENTRATION FOR COMPONENT',
     &I,' ',COMP(I)
      READ(9,*)U(I),C(I)
   41 CONTINUE
      WRITE(9,*)' INPUT A "1" IF THE SOLUBILITY CONTRAINT SHOULD BE'
      WRITE(9,*)' CHECKED FOR THE INITIAL GUESS'
      WRITE(9,*)' OR INPUT A "0" IF THE INITIAL GUESSES SHOULD BE'
      WRITE(9,*)' USED TO FIRST CALCULATE THE BATCH EQUILIBRIUM'
      WRITE(9,*)' WITHOUT THE SOLID.  THE SOLUBILITY CONSTRAINT'
      WRITE(9,*)' WILL THEN BE CHECKED AND ENFORCED IF NECESSARY'
      READ(9,*)IBATCH
C.....END OF DATA INPUT FROM KEYBOARD, USE THIS DATA TO CREATE AN
C.....INPUT FILE.  THE USER MAY THEN MODIFY DATA IN THE CREATED FILE
C.....AND CHOOSE DATA INPUT FROM FILE FOR OTHER TRIALS.
      WRITE(9,*)' ENTER A FILE NAME TO STORE INPUT DATA'
      READ(9,*)INDATA
      OPEN(8,FILE=INDATA,STATUS='NEW')
      WRITE(8,1000)TITLE
      WRITE(8,*)ALPHA,DELTA
      WRITE(8,*)NC,NCX,NS
C.....NAMES AND CONSTANTS
      DO 42 I=1,NC
   42 WRITE(8,*)COMP(I)
      DO 43 I=1,NCX
   43 WRITE(8,*)CMPLX(I),RK(I)
      DO 44 I=1,NS
   44 WRITE(8,*)PPT(I),RKSO(I)
C.....STOICHIOMETRIC MATRICES
C.....COMPLEXES
      DO 46 I=1,NCX
      WRITE(8,*)NCOMP(I)
      DO 45 J=1,NC
      IF(A(I,J) .NE. 0.0D0)WRITE(8,*)J,A(I,J)
   45 CONTINUE
   46 CONTINUE
C.....PRECIPITATES
      DO 48 I=1,NS
      WRITE(8,*)NPPT(I)
      DO 47 J=1,NC
      IF(B(I,J) .NE. 0.0D0)WRITE(8,*)J,B(I,J)
   47 CONTINUE
   48 CONTINUE
C.....TOTAL SOLUBLE COMPONENT CONCENTRATIONS AND INITIAL GUESSES
      DO 49 I=1,NC
   49 WRITE(8,*)U(I),C(I)
      WRITE(8,*)IBATCH
C
C.....END OF WRITING INPUT DATA TO FILE, SKIP NEXT SECTION WHICH
C.....READS INPUT DATA FROM A FILE
      GO TO 100
```

```
C
   50 CONTINUE
C.....READ INPUT DATA FROM A FILE
C.....GET DATA FILE NAME
      WRITE(9,*)' ENTER THE NAME OF THE INPUT DATA FILE'
      READ(9,*)INDATA
      OPEN(8,FILE=INDATA,STATUS='OLD')
      READ(8,1000)TITLE
      READ(8,*)ALPHA,DELTA
      READ(8,*)NC,NCX,NS
C.....NAMES AND CONSTANTS
      DO 51 I=1,NC
   51 READ(8,*)COMP(I)
      DO 52 I=1,NCX
   52 READ(8,*)CMPLX(I),RK(I)
      DO 60 I=1,NS
   60 READ(8,*)PPT(I),RKSO(I)
C.....STOICHIOMETRIC MATRICES
C.....SET ALL ENTRIES TO ZERO
      DO 65 I=1,NCX
      DO 65 J=1,NC
   65 A(I,J)=0.0D0
      DO 70 I=1,NS
      DO 70 J=1,NC
   70 B(I,J)=0.0D0
C.....COMPLEXES
      DO 80 I=1,NCX
      READ(8,*)NCOMP(I)
      DO 75 J=1,NCOMP(I)
      READ(8,*)JCOMP,JSTOIC
      A(I,JCOMP)=JSTOIC
   75 CONTINUE
   80 CONTINUE
C.....PRECIPITATES
      DO 90 I=1,NS
      READ(8,*)NPPT(I)
      DO 85 J=1,NPPT(I)
      READ(8,*)JCOMP,JSTOIC
      B(I,JCOMP)=JSTOIC
   85 CONTINUE
   90 CONTINUE
C.....TOTAL SOLUBLE COMPONENT CONCENTRATIONS AND INITIAL GUESSES
      DO 95 I=1,NC
   95 READ(8,*)U(I),C(I)
      READ(8,*)IBATCH
C
C.....END OF DATA INPUT SECTION
C
  100 CONTINUE
C
C.....BATCH EQUILIBRIUM PROGRAM
C
C.....SET OTHER NEEDED VARIABLES
      ITERMAX=50
C.....INITIALIZE VARIABLES
      DO 105 I=1,NC
      DO 104 J=1,NC
      RLHS(I,J)=0.0D0
      DF(I,J)=0.0D0
      DQ(I,J)=0.0D0
  104 CONTINUE
      F(I)=0.0D0
      Q(I)=0.0D0
  105 CONTINUE
C.....INITIALIZE ITERATION LOOP
      ITER=1
C.....RETURN HERE FOR ITERATIONS
  110 CONTINUE
C.....EVALUATE F(K) AND dF(K)/dC(J)
```

```
C.....FIRST ZERO VECTOR AND MATRIX
      DO 120 K=1,NC
      DO 115 J=1,NCX
  115 DF(K,J)=0.0D0
  120 F(K)=0.0D0
C.....F(K)
      DO 150 K=1,NC
      DO 150 I=1,NCX
      PROD=1.0D0
      DO 140 J=1,NC
  140 PROD=PROD*C(J)**A(I,J)
      F(K)=F(K)+A(I,K)*RK(I)*PROD
  150 CONTINUE
C.....DF(K,J)=dF(K)/dC(J)
      DO 160 K=1,NC
      DO 160 J=1,NC
      DO 160 I=1,NCX
      PROD=1.0D0
      DO 155 L=1,NC
  155 IF(L .NE. J)PROD=PROD*C(L)**A(I,L)
      DF(K,J)=DF(K,J)+A(I,J)*C(J)**(A(I,J)-1)*A(I,K)*RK(I)*PROD
  160 CONTINUE
C.....EVALUATE -G(I), THE RIGHT HAND SIDE OF MATRIX EQUATION
C.....STORE AS G(I)
      DO 170 I=1,NC
  170 G(I)=-(C(I)+F(I)-U(I))
C.....EVALUATE LEFT HAND SIDE FOR NEWTON-RAPHSON ITERATION
      DO 180 J=1,NC
      DO 180 K=1,NC
      IF(J .EQ. K)THEN
C.....ADD IDENTITY MATRIX
      RLHS(J,K)=1.0D0+DF(J,K)
      ELSE
      RLHS(J,K)=DF(J,K)
      ENDIF
  180 CONTINUE
C
C.....IF IBATCH = 0 SKIP PRECIPITATION SECTION AND SOLVE BATCH
C.....PROBLEM WITHOUT SOLID
      IF(IBATCH .EQ. 0)GO TO 275
C
C ***** THIS PORTION OF THE CODE USES THE PENALTY METHOD TO ENFORCE
C ***** THE SOLUBILITY PRODUCT, IT MAY BE ADDED TO EXISTING BATCH
C ***** EQUILIBRIUM PROGRAMS.
C.....CHECK SOLUBILITY CONSTRAINT
      PRECIP=.FALSE.
      DO 200 I=1,NS
      PROD=1.0D0
      DO 190 K=1,NC
  190 PROD=PROD*C(K)**B(I,K)
      Q(I)=PROD/RKSO(I)-1.0D0
      IF(Q(I) .GT. 0.0D0)PRECIP=.TRUE.
  200 CONTINUE
      IF(PRECIP)THEN
C.....MODIFY LHS MATRIX TO ENFORCE CONSTRAINT
C.....EVALUATE B(I)TRANSPOSE * dQ(K)/dC(J)
      DO 210 I=1,NC
      DO 210 J=1,NC
  210 DQ(I,J)=0.0D0
      DO 240 I=1,NC
      DO 230 J=1,NC
      DO 220 K=1,NS
  220 DQ(I,J)=DQ(I,J)+1./C(J)*B(K,I)*B(K,J)*(Q(K)+1)
      RLHS(I,J)=RLHS(I,J)+ALPHA*DQ(I,J)
  230 CONTINUE
  240 CONTINUE
C.....MODIFY G VECTOR TO ENFORCE CONSTRAINT(RECALL RHS IS -G(I))
C.....THE PENALTY TERM IS SUBTRACTED FROM THE RHS VECTOR
C.....-(G(I)+PENALTY) = -G(I)-PENALTY
```

```
C.....G(I)=G(I)-ALPHA*B(TRANSPOSE)*Q
      DO 250 K=1,NS
      DO 250 I=1,NC
  250 G(I)=G(I)-ALPHA*B(K,I)*Q(K)
      ENDIF
C***** END OF PENALTY FUNCTION ADDITIONS *****
C
  275 CONTINUE
C.....CALL SOLVER THAT PIVOTS AND SCALES TO SOLVE MATRIX EQUATION
C.....ANY SOLVER MAY BE USED HERE.
      CALL GAUSS(NC,RLHS,G)
C.....SOLUTION RETURNED IS THE CHANGE IN C
C.....UPDATE C VECTOR
      DO 290 I=1,NC
  290 C(I)=C(I)+G(I)
      IF(ITER .EQ. 1)THEN
C.....STORE DELTA C (0) TO USE FOR CONVERGENCE CRITERIA
      SUM=0.0
      DO 300 I=1,NC
  300 SUM=SUM+G(I)**2
      RNORM0=DSQRT(SUM)
      ELSE
C.....CHECK CONVERGENCE CRITERIA
      SUM=0.0
      DO 310 I=1,NC
  310 SUM=SUM+G(I)**2
      RNORM=DSQRT(SUM)
      CONV=RNORM/RNORM0
C.....IF CONVERGED GO TO 500
      IF(CONV .LT. DELTA)GO TO 500
      ENDIF
C.....INCREASE ITERATION COUNTER AND CHECK AGAINST ITERMAX
      ITER=ITER+1
      IF(ITER .GT. ITERMAX)THEN
      WRITE(9,*)' TOO MANY ITERATIONS'
      STOP
      ENDIF
      GO TO 110
C.....END OF ITERATION LOOP
C
C.....CONVERGED SOLUTION
  500 CONTINUE
C.....IF IBATCH = 0, THEN SET IBATCH = 1 AND RESOLVE TO CHECK
C.....SOLUBILITY CONSTRAINTS
      IF(IBATCH .EQ. 0)THEN
      IBATCH = 1
      GO TO 110
      ENDIF
C
C.....WRITE OUTPUT TO FILE 'OUTPUT', FILE NAME ENTERED AS FIRST DATA
C.....EVALUATE X AND P WITHIN OUTPUT LOOPS
C
      WRITE(10,1000)TITLE
      WRITE(10,1010)NC,NCX,NS,ALPHA,DELTA,ITER
C.....NAMES, CONSTANTS AND CONCENTRATIONS
      WRITE(10,1015)
      DO 902 I=1,NC
  902 WRITE(10,1020)I,COMP(I),U(I),C(I)
      WRITE(10,1025)
C.....COMPLEXES
      DO 903 I=1,NCX
      PROD=1.0D0
      DO 560 J=1,NC
  560 PROD=PROD*C(J)**A(I,J)
      X=RK(I)*PROD
  903 WRITE(10,1020)I,CMPLX(I),RK(I),X
C.....PRECIPITATES
WRITE(10,1030)
DO 904 I=1,NS
```

```
      IF(Q(I) .GT. 0.0D0)THEN
      P=ALPHA*Q(I)
      ELSE
      P=0.0D0
      ENDIF
  904 WRITE(10,1020)I,PPT(I),RKSO(I),P
C.....WRITE STOICHIOMETRIC MATRICES IN TABULAR FORM
C.....TABLES WITH EIGHT COMPONENTS ACROSS, ADDITIONAL TABLES ARE
C.....FORMED IF NC IS GREATER THAN EIGHT
C.....COMPLEXES
C.....SEE HOW MANY TABLES WILL BE NEEDED, ITABLE IS AN INTEGER
      ITABLE=NC/8+1
      ISTART=1
      IFIN=8
      WRITE(10,1031)
      DO 910 K=1,ITABLE
      IF(K .EQ. ITABLE)IFIN=NC
      WRITE(10,1035)(COMP(II),II=ISTART,IFIN)
      DO 905 I=1,NCX
      WRITE(10,1040)I,CMPLX(I),(A(I,J),J=ISTART,IFIN)
  905 CONTINUE
      ISTART=ISTART+8
      IFIN=IFIN+8
  910 CONTINUE
C.....PRECIPITATES
      ISTART=1
      IFIN=8
      WRITE(10,1032)
      DO 920 K=1,ITABLE
      IF(K .EQ. ITABLE)IFIN=NC
      WRITE(10,1036)(COMP(II),II=ISTART,IFIN)
      DO 915 I=1,NS
      WRITE(10,1040)I,PPT(I),(B(I,J),J=ISTART,IFIN)
  915 CONTINUE
      ISTART=ISTART+8
      IFIN=IFIN+8
  920 CONTINUE
C.....CLOSE FILES
      CLOSE(8)
      CLOSE(10)
C.....FORMAT STATEMENTS
 1000 FORMAT(20A4)
 1010 FORMAT(/5X,'NUMBER OF COMPONENTS',10X,I3,/,5X,
     1 'NUMBER OF COMPLEXES',11X,I3,/,5X,'NUMBER OF PRECIPITATES',
     2 8X,I3,/,5X,'PENALTY PARAMETER',15X,E10.4,/,5X,
     3 'CONVERGENCE CRITERIA',12X,E10.4,/,5X,
     4 'NUMBER OF ITERATIONS USED',6X,I2)
 1015 FORMAT(/5X,'COMPONENT',10X,'NAME',10X,
     1 'TOTAL SOLUBLE CONCENTRATION',5X,'CONCENTRATION')
 1020 FORMAT(7X,I3,15X,A8,10X,G10.4,20X,G10.4)
 1025 FORMAT(/6X,'COMPLEX',11X,'NAME',12X,'EQUILIBRIUM CONSTANT',
     1 10X,'CONCENTRATION')
 1030 FORMAT(/4X,'PRECIPITATE',9X,'NAME',12X,'SOLUBILITY PRODUCT',
     1 12X,'CONCENTRATION')
 1031 FORMAT(//20X,'STOICHIOMETRIC COEFFICIENTS FOR COMPLEXES',/,
     1 40X,'COMPONENTS')
 1032 FORMAT(//20X,'STOICHIOMETRIC COEFFICIENTS FOR PRECIPITATES',
     1 /,40X,'COMPONENTS')
 1035 FORMAT(6X,'COMPLEX',8X,'NAME',3X,8(2X,A8))
 1036 FORMAT(4X,'PRECIPITATE',6X,'NAME',3X,8(2X,A8))
 1040 FORMAT(7X,I3,10X,A8,2X,8(I2,8X))
      STOP
      END
      SUBROUTINE GAUSS(NEQ,A,F)
C.....SUBROUTINE TO PERFORM GAUSS ELIMINATION ON THE MATRIX EQUATION
C.....  A * X = F.  THE SOLUTION IS RETURNED IN THE LOADING
C.....VECTOR F, NEQ IS THE NUMBER OF EQUATIONS.  THE DIMENSIONS OF F
C.....AND A MUST MATCH THOSE IN THE CALLING PROGRAM.
C.....ADAPTED FROM NAIVE GAUSS ELIMINATION PROGRAM FROM
```

```
C.....CHAPRA AND CANALE (1985) SCALING AND PIVOTING ARE ADDED.
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(10,10),F(10)
      NM1=NEQ-1
C.....SCALE EQUATIONS
      DO 100 K=1,NEQ
      RMAX=0.0D0
      DO 50 J=1,NEQ
C.....FIND LARGEST ENTRY IN ROW K
      IF(DABS(A(K,J)) .GT. RMAX)RMAX=A(K,J)
   50 CONTINUE
      IF(RMAX .EQ. 0.0D0)THEN
      WRITE(9,*)' ROW',K,' HAS ALL ZEROS, CANNOT SOLVE'
      STOP
      ELSE
      DO 75 J=1,NEQ
      A(K,J)=A(K,J)/RMAX
   75 CONTINUE
      F(K)=F(K)/RMAX
      ENDIF
  100 CONTINUE
C.....FORWARD ELIMINATION
      DO 300 K=1,NM1
C.....PARTIALLY PIVOT ON PIVOT COLUMN K
      CALL PPIVOT(A,F,K,NEQ)
      KK=K+1
      DO 250 I=KK,NEQ
      C=A(I,K)/A(K,K)
      F(I)=F(I)-C*F(K)
      DO 240 J=KK,NEQ
      A(I,J)=A(I,J)-C*A(K,J)
  240 CONTINUE
  250 CONTINUE
      DO 260 I=KK,NEQ
  260 A(I,K)=0.0D0
  300 CONTINUE
C.....BACK SUBSTITUTION
      F(NEQ)=F(NEQ)/A(NEQ,NEQ)
      DO 350 NN=1,NM1
      SUM=0.0
      I=NEQ-NN
      IP1=I+1
      DO 320 J=IP1,NEQ
  320 SUM=SUM+A(I,J)*F(J)
      F(I)=(F(I)-SUM)/A(I,I)
  350 CONTINUE
      RETURN
      END
      SUBROUTINE PPIVOT(A,F,K,NEQ)
C.....SUBROUTINE TO PARTIALLY PIVOT MATRIX BY
C.....SEARCHING PIVOT COLUMN, K, FOR LARGEST ENTRY AND
C.....EXCHANGING ENTRIES TO MAKE IT THE PIVOT COEFFICIENT
C.....ADAPTED FROM CHAPRA AND CANALE (1985)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(10,10),F(10)
      KP1=K+1
C.....STORE PIVOT ROW AND ENTRY
      JJ=K
      B=DABS(A(K,K))
C.....COMPARE COLUMN ENTRIES
      DO 100 I=KP1,NEQ
      IF(DABS(A(I,K)) .GT. B)THEN
      B=A(I,K)
      JJ=I
      ENDIF
  100 CONTINUE
C.....IF INITIAL PIVOT ELEMENT WAS THE LARGEST, RETURN
      IF(K .EQ. JJ)RETURN
C.....SWITCH ROWS TO PUT LARGEST ENTRY IN PIVOT POSITION
```

```
      DO 200 J=K,NEQ
      TEMP=A(JJ,J)
      A(JJ,J)=A(K,J)
      A(K,J)=TEMP
  200 CONTINUE
      TEMP=F(JJ)
      F(JJ)=F(K)
      F(K)=TEMP
      RETURN
      END
```

Example Problem 1—Input

```
100000.,1.0E-05
 3 1 1
C1
C2
C3
C1C2      .8000
C2C3      6.0
 2
 1 1
 2 1
 2
 2 1
 3 1
2.600 0.7402
5.000 3.1403
4.400 4.4
1
```

Example Problem 1—Output

```
NUMBER OF COMPONENTS              3
NUMBER OF COMPLEXES               1
NUMBER OF PRECIPITATES            1
PENALTY PARAMETER                 0.1000E+06
CONVERGENCE CRITERIA              0.1000E-04
NUMBER OF ITERATIONS USED        10

 COMPONENT          NAME          TOTAL SOLUBLE CONCENTRATION     CONCENTRATION
     1               C1                    2.600                      1.000
     2               C2                    5.000                      2.000
     3               C3                    4.400                      3.000

  COMPLEX           NAME            EQUILIBRIUM CONSTANT           CONCENTRATION
     1              C1C2                 .8000                        1.600

PRECIPITATE         NAME            SOLUBILITY PRODUCT             CONCENTRATION
     1              C2C3                 6.000                        1.400


                STOICHIOMETRIC COEFFICIENTS FOR COMPLEXES
                                    COMPONENTS
  COMPLEX        NAME      C1        C2        C3
     1          C1C2       1         1         0


                STOICHIOMETRIC COEFFICIENTS FOR PRECIPITATES
                                    COMPONENTS
PRECIPITATE      NAME      C1        C2        C3
     1           C2C3       0         1         1

NUMBER OF COMPONENTS              3
NUMBER OF COMPLEXES               1
NUMBER OF PRECIPITATES            1
PENALTY PARAMETER                 0.1000E+06
CONVERGENCE CRITERIA              0.1000E-04
NUMBER OF ITERATIONS USED        10
```

COMPONENT	NAME	TOTAL SOLUBLE CONCENTRATION	CONCENTRATION
1	C1	2.600	1.000
2	C2	5.000	2.000
3	C3	4.400	3.000

COMPLEX	NAME	EQUILIBRIUM CONSTANT	CONCENTRATION
1	C1C2	.8000	1.600

PRECIPITATE	NAME	SOLUBILITY PRODUCT	CONCENTRATION
1	C2C3	6.000	1.400

STOICHIOMETRIC COEFFICIENTS FOR COMPLEXES

		COMPONENTS		
COMPLEX	NAME	C1	C2	C3
1	C1C2	1	1	0

STOICHIOMETRIC COEFFICIENTS FOR PRECIPITATES

		COMPONENTS		
PRECIPITATE	NAME	C1	C2	C3
1	C2C3	0	1	1

Example Problem 2—Input

```
100.,1.0E-05
 3 6 1
CA
H+
CO3
CACO3     1.000E+3
CAHCO3+   3.98107E+11
CAOH+     6.30957E-13
HCO3-     1.58489E+10
H2CO3     3.16228E+16
OH-       1.000E-14.0
CACO3     5.01187E-09
 2
 1 1
 3 1
 3
 1 1
 2 1
 3 1
 2
 1 1
 2 -1
 2
 2 1
 3 1
 2
 2 2
 3 1
 1
 2 -1
 2
 1 1
 3 1
1.0000E-03 6.87068E-04
.0000      3.86367E-11
1.0000E-03 4.32514E-04
```

Example Problem 2—Output

NUMBER OF COMPONENTS	3
NUMBER OF COMPLEXES	6
NUMBER OF PRECIPITATES	1
PENALTY PARAMETER	0.1000E+03
CONVERGENCE CRITERIA	0.1000E-04
NUMBER OF ITERATIONS USED	7

COMPONENT	NAME	TOTAL SOLUBLE CONCENTRATION	CONCENTRATION
1	CA	0.1000E-02	0.1216E-03

2	H+	.0000	0.1240E-09
3	CO3	0.1000E-02	0.4122E-04

COMPLEX	NAME	EQUILIBRIUM CONSTANT	CONCENTRATION
1	CACO3	1000.	0.5012E-05
2	CAHCO3+	0.3981E+12	0.2474E-06
3	CAOH+	0.6310E-12	0.6188E-06
4	HCO3-	0.1585E+11	0.8099E-04
5	H2CO3	0.3162E+17	0.2003E-07
6	OH-	0.1000E-13	0.8066E-04

PRECIPITATE	NAME	SOLUBILITY PRODUCT	CONCENTRATION
1	CACO3	0.5012E-08	0.8327E-03

STOICHIOMETRIC COEFFICIENTS FOR COMPLEXES

		COMPONENTS		
COMPLEX	NAME	CA	H+	CO3
1	CACO3	1	0	1
2	CAHCO3+	1	1	1
3	CAOH+	1	-1	0
4	HCO3-	0	1	1
5	H2CO3	0	2	1
6	OH-	0	-1	0

STOICHIOMETRIC COEFFICIENTS FOR PRECIPITATES

		COMPONENTS		
PRECIPITATE	NAME	CA	H+	CO3
1	CACO3	1	0	1