JOURNAL OF COMPUTATIONAL PHYSICS 60, 219–234 (1985)

Performance of Algorithms for Calculating the Equilibrium Composition of a Mixture of Gases

KEITH MEINTJES AND ALEXANDER P. MORGAN

General Motors Research Laboratories,
Warren, Michigan 48090-9055

Received November 18, 1983; revised December 7, 1984

Some strategies for calculating the effects of chemical reactions in large, multidimensional finite-difference computer codes require the equilibrium gas composition as an input. Because the equilibrium problem must be solved a large number of times, it is essential that the equilibrium solver be fast and reliable. An existing solver is a variant of the Gauss–Seidel technique, and its performance can be predicted and quantified. It is relatively fast, but can be unreliable. By contrast, Newton's method is slower but more reliable. A hierarchical algorithm, in which recourse is made to Newton's method if Gauss–Seidel iteration fails, is shown to combine the speed of Gauss–Seidel and the reliability of Newton. The hierarchical solver has been incorporated into the CONCHAS computer code. The reliability of the code is improved, and there is a decrease in the amount of computer time required. The new algorithm has not failed during production runs of CONCHAS, but it has failed to find the solution of some special test problems. © 1985 Academic Press, Inc.

1. INTRODUCTION

For calculations of chemically reactive fluid flow it is often advantageous to classify chemical reactions as "fast" or "slow," and then to consider the fast reactions to be in chemical equilibrium. This concept of "partial equilibrium flow," described by Ramshaw [1], allows a realistic treatment of post-flame conditions during calculations of engine combustion. Such calculations would be impractical if the post-flame chemistry were to be calculated using kinetic mechanisms, but are practical if fast equilibrium solvers can be used.

Another approach which requires a fast equilibrium solver is that of Reitz and Bracco [2, 3]. They have introduced a "global local equilibrium kinetics model" for flows involving combustion, in which a global reaction rate is used to determine the rate at which chemical equilibrium is attained. This approach is fundamentally different from partial equilibrium, but also requires the local equilibrium composition as an input.

In this paper we discuss the problem of finding the equilibrium composition of a mixture of gases, mainly from the point of view of multidimensional calculations of combustion in engines and the CONCHAS family of computer codes [4, 5].

0021-9991/85 $3.00

The primary requirement of an equilibrium solver in a large finite-difference code is that it be fast. A typical run of CONCHAS requires the equilibrium problem to be solved about a million times. Typically, the speed of an algorithm in solving a single problem is measured in milliseconds. Each extra millisecond required by the solver will translate into about 17 additional min of computer CPU time.

There are various computer codes available for calculating the equilibrium composition of a mixture of species, the best known being the NASA code [6]. Olikara and Borman [7] developed a code primarily for zero-dimensional (thermodynamic) models with hydrocarbon combustion. However, these codes are too slow to be used for the CONCHAS equilibrium calculation. They also do not exploit the special structure of the CONCHAS problem, which, in particular, involves a reduced-order system of equations and, generally, good initial solution estimates.

Ramshaw and Cloutman [8] developed the equilibrium solver in the original version of the CONCHAS code. They found the solver to converge quickly and to be more effective than Newton iteration. However, under certain conditions, their algorithm fails because it converges extremely slowly. ("Failure" is defined as the inability to reach a solution within a specified number of iterations.) In this paper the reasons for this failure are identified and it is shown that a hierarchical solver, in which recourse is made to Newton's method if the original algorithm fails, runs about as fast as the original and is more reliable.

2. FORMULATION

The equilibrium composition must be found for each cell at each time step. We accept the following: The equilibrium calculation for each computational cell is independent from that of its neighbors and may be treated in isolation, and for each cell the equilibrium composition is to be found for a constant-volume system of specified temperature and mass. (These conditions are *not* assumptions, but consequences of the solution method used for the overall governing equations [1, 8].)

In general, if there are N distinct chemical species composed of M chemical elements, then the algebraic system to be solved consists of $N-M$ nonlinear equilibrium equations and M linear element-conservation relations.

If X represents a mole of any species, the equilibrium reactions are of the form

$$\sum_k a_{ks} X_k \rightleftarrows \sum_k b_{ks} X_k \qquad (1)$$

where $k=1,\ldots,N$, $s=1,\ldots,N-M$ and a_{ks} and b_{ks} are the dimensionless stoichiometric coefficients. The equilibrium reactions lead to algebraic relations of the form

$$K_s R_s = P_s \qquad (2)$$

where the products

$$R_s = \prod_k [c_k]^{a_{ks}} \qquad \text{and} \qquad P_s = \prod_k [c_k]^{b_{ks}} \tag{3}$$

are respectively associated with the "reactants" (left-hand side) and the "products" (right-hand side) of Eq. (1). In the above equations K_s is the equilibrium constant for reaction s, while c_k is the molar concentration of species k. Finally, if we adopt the index v for the chemical elements, then the linear element-conservation reactions are of the form

$$\sum_k d_{kv} c_k = C_v \qquad (v = 1, ..., M). \tag{4}$$

Information on element conservation is, however also contained in the equilibrium reactions themselves. For example, the reaction $2\,OH \rightleftarrows O_2 + H_2$ implies: "Two moles of OH can be interchanged with one mole of O_2 and one of H_2 without violating atom conservation."

This fact allows us to construct iterative methods for solving the equilibrium problem which explicitly involve only the $N - M$ nonlinear relations, Eqs. (2). In the above example, a change of $\delta O_2 = \varepsilon$ in the O_2 concentration should be accompanied by changes of $\delta H_2 = \varepsilon$ and $\delta OH = -2\varepsilon$ in the H_2 and OH concentrations, respectively. In the general notation, this may be written as

$$(\delta c_k)_s = (b_{ks} - a_{ks})\, \varepsilon_s. \tag{5}$$

Of course, a reduced system can be obtained by substituting the linear equations directly into the nonlinear. It can be shown that the following algebraic systems are mathematically equivalent:

- The full $N \times N$ system, Eqs. (2) and (4).
- The (reduced) $(N - M) \times (N - M)$ system of Eq. (2), complemented by the correction constraints of Eq. (5).
- A reduced system obtained by substituting the M linear relations, Eq. (4), into the nonlinear, Eq. (2), i.e., an $(N - M) \times (N - M)$ system.

Two principal methods will be considered below to solve the equilibrium problem: Newton's method and the Gauss–Seidel method [9]. As will be observed, the original method of solution is a Gauss–Seidel variant. It can also be shown that:

- Newton's method applied to any of the formulations will give exactly the same sequence of iterates.
- The Gauss–Seidel method applied to either of the reduced $(N - M) \times (N - M)$ formulations will result in exactly the same sequence of iterates.

Consequently, we have the option of applying either iterative technique to a system of $N - M$ equations rather than the larger equivalent $N \times N$ system.

Now let us observe that there are three issues connected with using any local method of solution. (A "local" method is one which requires an initial guess and which will converge to a particular solution when the initial guess is sufficiently close to that solution.)

- Will it converge?
- How fast will it converge?
- How are the initial guesses to be chosen?

These questions are not independent. For example, the choice of initial guess may affect the speed of convergence. Also note that very slow convergence is, for practical purposes, the same as non-convergence.

In this connection, it is important to distinguish the *local* and the *global* characteristics of a solution method. The local convergence rate, when a method is close to and converging to a solution, is independent of the initial values and can often be quantified. Little can be said (mathematically) about global convergence—whether a given set of initial values will converge to a particular solution, or the number of iterations needed for convergence. However, a poor rate of local convergence obviates considerations of global behavior. It is possible to affect global behavior within an algorithm, for example, by restricting the iterates to be positive when the solution is known to be positive. In practice, however, the problem of global convergence is addressed most directly by seeking to provide good initial solution estimates.

Newton's method is commonly used to solve systems of algebraic equations. Its convergence characteristics are generally better than those of the Gauss–Seidel method, in that it tends to be more reliable and takes fewer iterations. However, the Gauss–Seidel method has one possible advantage over Newton's method—the speed of a single iteration. That is, if both methods are equally reliable (in a particular setting) and both methods take a comparable number of iterations, then the Gauss–Seidel method will be faster. This is particularly significant for very large systems, say, systems with hundreds of variables.

We now concentrate on methods for solving the reduced system of $N-M$ equations. A particular iterative solution method is characterized by the way in which the corrections (the ε of Eq. (5)) are evaluated and applied to the individual species.

The Gauss–Seidel method considers each equation (reaction) in isolation from the others. Gauss–Seidel variants are distinguished by the manner in which the individual equations are solved. For example, in the one-step Gauss–Seidel–Newton method, each equation in turn is updated by a single Newton iterate. Specifically, if Eq. (2) is not satisfied for some s, we may write it as

$$E_s\{c_k\} = K_s R_s\{c_k\} - P_s\{c_k\} \tag{6}$$

and seek a correction ε_s to drive E_s to zero. Then

$$E_s\{c_k + \delta c_{ks}\} = E_s\{c_k\} + \sum_i \left[(\partial E_s/\partial c_k)\,\delta c_{ks}\right] + \cdots \tag{7}$$

or, setting the updated value of $E_s = 0$ and truncating the series,

$$0 = E_s\{c_k\} + \sum_k [(\partial E_s/\partial c_k)(b_{ks} - a_{ks})]\, \varepsilon_s. \tag{8}$$

This equation is to be solved for ε, yielding a Newton iteration on the single equation being considered. The species concentrations are updated according to Eq. (5), after which a correction is found for the next equation, and so on.

A full Newton's method for the system is obtained by solving the matrix equation

$$\mathbf{J} \cdot \boldsymbol{\varepsilon} + \mathbf{E}\{c_k\} = \mathbf{0} \tag{9}$$

for the vector of corrections, **ε**. **J** is the Jacobian matrix, whose element in the sth row and nth column is given by

$$J_{sn} = \sum_k [(\partial E_s/\partial c_k)(b_{kn} - a_{kn})]. \tag{10}$$

3. The Gauss–Seidel Method

The algorithm of Ramshaw and Cloutman [8] solves each equation sequentially and is thus a Gauss–Seidel variant. The novel feature of this method is a quadratic evaluation of the correction, ε, to be applied to each equation. To first order the correction is a single Newton iteration, and when ε is small, i.e., the method is close to a solution, the quadratic aspect can be expected to be less important. Indeed, we have found that the details of the quadratic evaluation do not materially affect the local convergence rate of the method. (The original paper [8] and subsequent publications do not observe that that algorithm is a Gauss–Seidel variant.)

The basic Gauss–Seidel method and some of its variants have, in fact, been carefully studied, and its local behavior is well understood (see Ortega and Rheinholdt [9]). In particular, there is a theorem that predicts the local convergence rate of the method. The essential details are summarized below.

Let $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ denote a system of n equations

$$f_i(x_1, ..., x_n) = 0$$

in n unknowns, for $i = 1, ..., n$. The Gauss–Seidel method begins at an "initial guess," $\mathbf{x}^{(0)}$, for a solution to $\mathbf{f} = \mathbf{0}$ and then improves the guess iteratively with $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$,..., being better and better estimates.

A Gauss–Seidel iteration is in two steps:

- Step 1. Solve for x_i:

$$f_i(x_1^{(\kappa+1)}, ..., x_{i-1}^{(\kappa+1)}, x_i, x_{i+1}^{(\kappa)}, ..., x_n^{(\kappa)}) = 0. \tag{11}$$

Step 2. Set $x_i^{(\kappa+1)} = x_i$.

Note that in solving the one-variable problem in Step 1 of each iteration, we will generally have to choose one solution out of several solutions to the polynomial, Eq. (11). Usually, one chooses the closest solution to $x_i^{(\kappa)}$.

Let $\mathbf{J}(\mathbf{x})$ denote the Jacobian matrix. We can write this matrix as a sum of parts as

$$\mathbf{J}(\mathbf{x}) = \mathbf{D} - \mathbf{L} - \mathbf{U} \tag{12}$$

where $\mathbf{D}$ is the diagonal, $-\mathbf{L}$ the lower triangle, and $-\mathbf{U}$ the upper triangle of the Jacobian. Assume that $\mathbf{D}$ has no zero entries, and define

$$\mathbf{H} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{U}. \tag{13}$$

We will be interested in the spectral radius of this matrix (i.e., the maximum of the absolute values of the eigenvalues) evaluated at a point x, denoted $\rho = \rho\{\mathbf{H}(\mathbf{x})\}$. Now we have this result, which is an immediate corollary to the nonlinear SOR theorem [9, p. 326]:

THEOREM. *If* $\mathbf{x}^*$ *is a solution to* $\mathbf{f}(\mathbf{x}) = 0$ *and* $\rho\{\mathbf{H}(\mathbf{x}^*)\} < 1$, *then, for initial guesses close enough to* $\mathbf{x}^*$, *the Gauss–Seidel method converges to* $\mathbf{x}^*$.

Further, for any $\sigma > 0$ *with* $\rho + \sigma > 1$ *and each initial point* $\mathbf{x}^{(0)}$ *in the region of convergence around* $\mathbf{x}^*$, *there is a* $\kappa^0 = \kappa^0(\mathbf{x}^{(0)}, \sigma)$ *such that*

$$|\mathbf{x}^{(\kappa)} - \mathbf{x}^*| \leqslant (\rho + \sigma)^\kappa \qquad \text{for } \kappa \geqslant \kappa^0, \tag{14}$$

where $\mathbf{x}^{(\kappa)}$ *denotes the* κ*th Gauss–Seidel iterate beginning at* $\mathbf{x}^{(0)}$. *The number* ρ *is the smallest positive number with this property.*

For Gauss–Seidel iteration, the theorem identifies the spectral radius of the $\mathbf{H}$ matrix as controlling the behavior of the algorithm, The equality of Eq. (14) holds for the worst choices of $\mathbf{x}^{(0)}$. Other values of $\mathbf{x}^{(0)}$ will give faster convergence, and the theorem implies that σ can be arbitrarily small if κ^0 is sufficiently large. Then $-1/\log(\rho)$ is an upper bound on the number of iterations required to gain an additional significant digit in the solution, once the solution estimates are in the region of convergence (cf. [10, p. 64]). The theorem does not indicate how to find an initial guess, but focuses on the local behavior of the algorithm when it is close to, and converging to, a solution.

As stated above, the theorem implies that the one-variable system, Eq. (11), is to be solved exactly. The theorem holds, essentially unchanged, for an m-step Gauss–Seidel–Newton method in which the (exact) solution of Eq. (11) is replaced by m Newton iterates [9, p. 327].

The fact that (with $m = 1$) taking one step of Newton's method is just as good as solving Eq. (11) exactly suggests that there is no need to solve each individual equation very accurately. The Gauss–Seidel aspect, that of solving individual equations sequentially, dominates the rate of convergence.

The actual algorithm used in CONCHAS differs from the standard one-step Gauss–Seidel–Newton method in two ways:

- It does not allow any iterate to go negative.
- It uses a quadratic evaluation of the correction ε.

The first modification is motivated by the desire to find the correct, physical solution in which all the concentrations are positive. It can be regarded as a way of getting a better intitial guess to restart the method, and has no effect on the local convergence rate.

The "quadratic" evaluation of the correction is also based on the global information that the physically meaningful solution must be positive, and not on the continuation of the series expansion of Eq. (7). The variants of the theorem noted above suggest that this modification does not materially change the local convergence characteristics of the method. Indeed, we have found that these modifications affect the global and not the local behavior of the method.

The theorem has a number of important implications for the Gauss–Seidel method and its variants.

- High-order accuracy in solving the one-variable problem is not important in determining the local convergence rate.
- The limiting convergence rate is always linear, i.e., the ratio of the error on two successive iterations is a constant. (By contrast, Newton's method generally converges quadratically, i.e., the error follows the sequence 10^{-1}, 10^{-2}, 10^{-4}, 10^{-8},..., and, if the accuracy requirement is stringent enough, the Newton method will converge faster than the Gauss–Seidel method).
- The rate of convergence is affected by the order in which the equations are solved, since this order affects the decomposition of the Jacobian, Eq. (12).
- The rate of convergence is affected by the choice of equilibrium reactions, since different choices of reactions will result in different Jacobians.

4. APPLICATION–HYDROCARBON COMBUSTION

As formulated in the standard CONCHAS codes, the partial equilibrium concept requires the calculation of the equilibrium composition of the ten species shown in Table I. (The fuel, subscript 1, does not enter into the equilibrium problem.)

The equilibrium reactions should be written to represent the formation of chemical compounds from elements in their standard states [11]. This ensures an independent set of equations. (However, non-standard reactions, formed by linear combinations of the standard reactions, may be used as input to a particular algorithm.) We choose CO as a standard state, since C (graphite) is not included in the calculation. The equilibrium relations (cf., Eq. (2)) and the corresponding nonlinear algebraic equations (Eq. (3)) are given in Table II as (a)–(f).

Ramshaw and Cloutman found their algorithm to converge faster when reaction (f) was replaced by (f′), which is a combination of (e) and (f).

TABLE I

Species Considered

Subscript	Species	
2	O_2	Molecular oxygen
3	N_2	Molecular nitrogen
4	CO_2	Carbon dioxide
5	H_2O	Water vapor
6	H	Atomic hydrogen
7	H_2	Molecular hydrogen
8	O	Atomic oxygen
9	N	Atomic nitrogen
10	OH	Hydroxyl radical
11	CO	Carbon monoxide

The element-conservation equations

$$C_O = 2c_2 + 2c_4 + c_5 + c_8 + c_{10} + c_{11} \tag{g}$$

$$C_H = 2c_5 + c_6 + 2c_7 + c_{10} \tag{h}$$

$$C_C = c_4 + c_{11} \tag{i}$$

$$C_N = 2c_3 + c_9 \tag{j}$$

are not explicitly used.

The solutions to the equilibrium problem are best expressed in terms of three physical variables: the temperature T, the pressure p (at the solution), and the equivalence ratio Φ. The pressure, which reflects the total system density (mass per unit volume), is given by

$$p = \sum_k (c_k \mathscr{R} T) \tag{15}$$

where $\mathscr{R} = 8.4143 \times 10^7$ g cm^2/(g-mole s^2 K) is the universal gas constant. The units

TABLE II

Equilibrium Expressions

$N_2 \rightleftarrows 2\,N$	$K_1 c_3 = (c_9)^2$	(a)
$O_2 \rightleftarrows 2\,O$	$K_2 c_2 = (c_8)^2$	(b)
$2\,CO_2 \rightleftarrows 2\,CO + O_2$	$K_3 (c_4)^2 = c_2 (c_{11})^2$	(c)
$H_2 \rightleftarrows 2\,H$	$K_4 c_7 = (c_6)^2$	(d)
$2\,OH \rightleftarrows O_2 + H_2$	$K_5 (c_{10})^2 = c_7 c_2$	(e)
$2\,H_2O \rightleftarrows 2\,H_2 + O_2$	$K_6 (c_5)^2 = (c_7)^2 c_2$	(f)
$2\,H_2O + O_2 \rightleftarrows 4\,OH$	$K_6' (c_5)^2 c_2 = (c_{10})^4$	(f′)

of pressure are dyn/cm^2 (1 Pa = 10 dyn/cm^2), and the concentrations are expressed in g-mole/cm^3. The molar fuel–air equivalence ratio

$$\Phi = (4C_C + C_H)/(2C_O) \tag{16}$$

expresses the relative amounts of fuel (which contains carbon and hydrogen, contributing to the numerator) and air (containing oxygen and contributing to the denominator) present.

In running the CONCHAS code, we had found that the equilibrium calculation would, under certain circumstances, fail to converge. This generally occurred for fuel-rich combustion ($\Phi > 1$), and during the expansion stroke as the temperature decreased. The reactions used where (a) through (e) and (f′) of Table II.

To verify the results of the theorem and to identify why the original algorithm was failing, a set of some 500 problems was generated during a run of CONCHAS. Since the code generally provides a good initial estimate of the solution, these problems were readily solved (to machine accuracy) by Newton iteration. With the solution known, the spectral radius of the matrix **H**, Eq. (13), could be found and the error behavior of the Gauss–Seidel algorithm could be monitored.

The results are shown in Fig. 1. The "actual eigenvalue" was calculated from the Jacobian of the system evaluated at the solution and represents the prediction of the theorem, while the "computed eigenvalue" was obtained from the observed rate of convergence through Eq. (14) and reflects the measured performance of the method. It is clear that the variant of Gauss–Seidel used in CONCHAS behaves as predicted by the theorem.

These computations also reveal the conditions under which the Gauss–Seidel algorithm fails. Figure 2 shows that, for the problems considered, the eigenvalue is close to unity and convergence is extremely slow when the equivalence ratio $\Phi \geqslant 1$.

To ascertain the effect of the quadratic feature of the method of [8], these computations were repeated using a one-step Gauss–Seidel–Newton method. The local

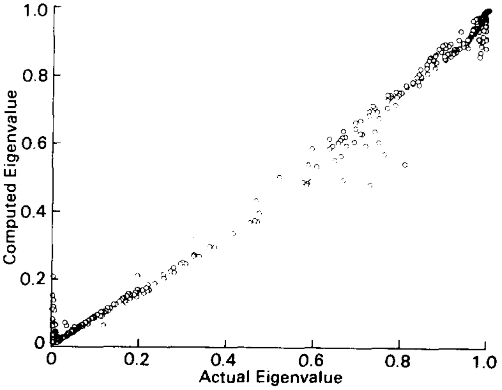

FIG. 1. Actual and computed eigenvalues.

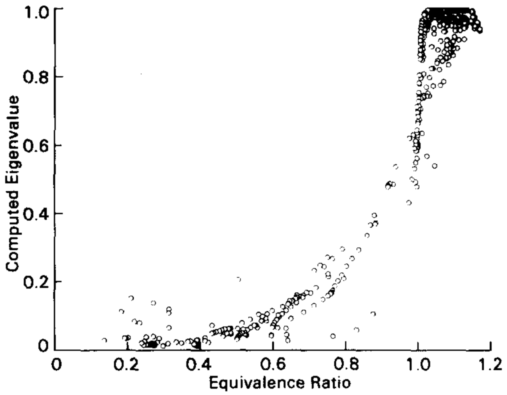

FIG. 2. Dependence of computed eigenvalue on equivalence ratio.

convergence rates of the Gauss–Seidel–Newton method and the quadratic method were essentially the same, confirming the expectation that the quadratic feature does not affect the local behavior of the CONCHAS algorithm. The quadratic method succeeded in solving some problems on which the one-step Gauss–Seidel–Newton method failed, indicating better global behavior. No attempt was made to improve the global performance of the Gauss–Seidel–Newton method.

The theorem was then used to predict the performance of the Gauss–Seidel methods over the range of variation of the physical variables affecting the problem. The spectral radius is relatively insensitive to both the pressure and the hydrogen-to-carbon ratio of the fuel, and is affected mainly by the temperature and the equivalence ratio. The influence of these latter variables is shown in Fig. 3.

The correspondence of the predictions of the theorem and the observed behavior of the Gauss–Seidel method indicate that the algorithm was failing in the CONCHAS code primarily because of poor local convergence and not for global reasons such as poor start points. The initial estimate used in CONCHAS (which is a perturbation of the previous solution) is usually good.

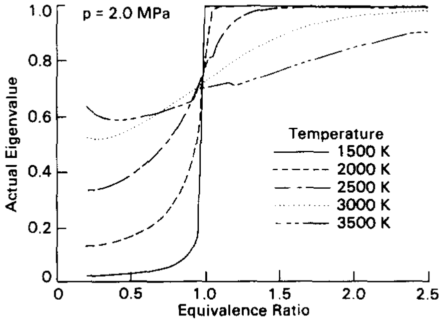

FIG. 3. Dependence of actual eigenvalue on physical conditions.

5. Selection of the Equilibrium Reactions

The theorem effectively predicts the local performance of the Gauss–Seidel methods and provides a means to ascertain the effect of different choices of equilibrium reactions on the local convergence rate. This was done for the present problem of hydrocarbon combustion, and the results are briefly described below. Unfortunately, this effort did not lead to general guidelines which would facilitate an a priori choice of reactions to be used.

The speed of convergence of the Gauss–Seidel method is determined by the spectral radius of **H**, Eq. (13). This spectral radius can be changed in two ways:

- By altering the order in which the equations are solved.
- By changing the equations through algebraic manipulations.

Both strategies were used by Ramshaw and Cloutman to make the algorithm work better; indeed, we found that, at least for $\Phi < 1$, the spectral radius when reaction (f′) is used is smaller than that for reaction (f).

A set of solutions was generated for a range of temperature, pressure, and equivalence ratio. The Jacobian and its spectral radius were then calculated for different sets of equilibrium reactions. No system of equations could be found for which the Gauss–Seidel method would work "well" over the entire range of equivalence ratio and temperature, and we have accordingly used the same reactions as in [8] for all our computations.

Although one can make some intuitive observations about why the Gauss–Seidel method fails, there seems to be no systematic way of choosing the reactions for optimal performance. Obviously, one goal might be to make the Jacobian "diagonally dominant," or to reduce the coupling between the reactions. It is, however, difficult to analyze the effect of different reactions on the off-diagonal terms. Even when the spectral radius of **H** is small, the Jacobian is not necessarily diagonally dominant, and the reactions are not decoupled in any obvious way.

Changing the order in which the equations are solved is equivalent to row and column interchanges of the Jacobian. When this was done, the spectral radius was found to be less affected than by changing the reactions.

Thus, the theorem provides a way to predict the performance of the Gauss–Seidel methods with a particular set of equilibrium reactions. However, there is no way to identify a priori an "optimum" set of reactions. Since the performance is greatly affected by the reactions used, any description of the Gauss–Seidel method (or variants thereof) is incomplete without details of the reaction set used and the order in which the equations are solved.

An interesting fact is that in none of the cases analyzed was the Gauss–Seidel method divergent; the spectral radius found from the Jacobian at the solution was always less than (but often very close to) unity. Thus the method was observed to always be locally convergent.

Computational tests of the quadratic method [8] showed a similar behavior (although these tests include the effect of the global modifications). The error tends always to decrease, and the final result is "closer" to the solution than the initial guess even when the method has failed by not satisfying an error criterion after a hundred iterations. At each iteration the solution is physical, i.e., all the concentrations are positive. Therefore, when the method "fails" by not satisfying an accuracy condition after a hundred iterations, the solution is inaccurate but may be physically reasonable. Failure of the equilibrium calculation does not cause an obvious "failure" of CONCHAS but does introduce inaccuracies in the overall computation. By improving the reliability of the equilibrium solver (by reducing failures) we improve the reliability of CONCHAS in the sense that the overall computed results then conform more closely to the physical assumptions made in the code.

6. A MORE RELIABLE "HIERARCHICAL" SOLUTION METHOD

The speed of the Gauss–Seidel method is an advantage which is difficult to disregard. Newton's method is more reliable, but requires more computational effort. We will show that a hierarchy, in which Newton is used to "back up" Gauss–Seidel, combines the desirable features of both methods.

To compare the methods, a set of 1000 problems was used. These were constructed by using a random number generator to select values of the temperature ($1300 \leqslant T \leqslant 3500$ K), pressure ($10^5 \leqslant p \leqslant 5 \times 10^6$ Pa) and equivalence ratio ($0.2 \leqslant \Phi \leqslant 2.3$) for each problem. The "distance," μ, of a point $\mathbf{c}$ from the solution $\mathbf{c}^*$ is defined by

$$\mu = \frac{|\mathbf{c} - \mathbf{c}^*|}{|\mathbf{c}^*|} \qquad \text{where } |\mathbf{c}| = \sqrt{\sum_k (c_k)^2}. \tag{17}$$

For the 1000 problems the initial guesses were all in the range $0.02 \leqslant \mu \leqslant 0.10$. This set of problems both covers the range of physical conditions encountered in calculations of engine combustion and simulates the type of problem encountered in CONCHAS with respect to the accuracy of the initial values. The problems were not screened in any way based on the performance of a particular method.

The linear system of equations to be solved for the vector of corrections in Newton's method has already been given, Eq. (9). We used a standar subroutine, LEQT2F, from the IMSL Library [12] to do this. "Failure" of the Newton iteration is indicated by an increase in the error norm. This is a very stringent requirement, demanding essentially a monotonic decrease in the error norm as the solution is approached. We did not investigate the reasons for failure in detail or attempt to restart the method with a better initial guess. However, the failures do not correlate with any physical variable (temperature, pressure, or equivalence ratio), and there is no region encompassing particular values of the physical variables in which Newton's method will tend to fail more often.

With respect to the discussion of the previous section, Newton's method does not depend at all on the order in which the equations are written, and we found that it is insensitive to the reactions used; it converges as quickly with either of the reactions (f) or (f′).

Computational tests of the original Gauss–Seidel algorithm and Newton's method showed the following:

- Gauss–Seidel is faster than Newton when it succeeds (usually for $\Phi < 1$). In fact, the original algorithm was timed at 0.75 ms per iteration, while Newton takes 7.3 ms/iteration, about 10 times longer. (These timings are for an IBM 370/3033.)
- Newton's method is more reliable than the Gauss–Seidel method, since it fails less often.

The speed of the Gauss–Seidel method and the reliability of Newton's method can be combined when a hierarchy is used: If the Gauss–Seidel method fails, recourse is made to Newton's method, with the Newton method using the final Gauss–Seidel result as a starting point.

The three methods were compared on the test set of 1000 problems and the results are shown in Table III. (Recall that a single Newton iteration is roughly equivalent to ten Gauss–Seidel iterations.)

With respect to the original algorithm and Newton's method, Table III shows the following:

- Increasing the accuracy requirement results in a large increase in the number of iterations required by the Gauss–Seidel method and also increases the number of failures. This is because the method is linearly convergent (see Eq. (14)). Newton's method is quadratically convergent and requires fewer additional iterations for increased accuracy.
- Newton's method is more reliable than the Gauss–Seidel method.
- Both methods take longer, on average, to fail than they do to find a solution.

The iteration data presented in Table III for the hierarchy concern only that subset of the 1000 problems which the Gauss–Seidel method failed to solve. For example, when $\mu \leqslant 10^{-2}$, the second level of the hierarchy solved $(938-538)=400$ problems. These 400 problems required an average 2.60 Newton iterations, in addition to the 100 iterations for the Gauss–Seidel method to fail. There were $1000-938=62$ problems which the hierarchy failed to solve. Note that a failure of the hierarchy requires about 20 times the computational effort of a successful Gauss–Seidel solution.

These 62 failures are not an indication of the reliability of the hierarchy in practice, as will be discussed below. The important observation is that more stringent accuracy requirements cause only a small increase in the number of times the hierarchy fails; however, the second-level Newton method is used more often as the number of first-level Gauss–Seidel failures increases. The hierarchy is the most reliable of the three methods.

TABLE III

Comparative Performance of Three Solvers

Error criterion $\mu \leqslant$	Original Gauss–Seidel method			Newton's method			Hierarchical solver		
	Successes		Failures	Successes		Failures	Successes		Failures
	No. of problems[a]	Average iterations	Average iterations	No. of problems[a]	Average iterations	Average iterations	No. of problems[a]	Average Newton iterations[b]	Average Newton iterations[b]
2×10^{-2}	574	8.57	100	859	2.30	14.62	940	2.26	13.82
10^{-2}	538	11.22	100	855	2.84	14.36	938	2.60	13.44
10^{-3}	475	19.47	100	845	4.04	15.57	936	3.30	13.17
10^{-4}	436	26.23	100	842	4.73	17.18	934	3.66	14.05
10^{-5}	394	28.72	100	841	5.22	17.70	933	3.78	15.33
10^{-6}	368	32.13	100	840	5.56	18.21	932	3.95	16.57

[a] Out of 1000 problems.

[b] Excluding Gauss–Seidel iterations.

The error criterion in the original CONCHAS code is approximately $\mu \leqslant 2 \times 10^{-2}$. This relatively weak condition, which ensures only two digits of accuracy, is predicated by the poor convergence characteristics of the Gauss–Seidel method and cannot be strengthened without causing the method to fail more often. The hierarchy allows a more stringent error criterion without a loss of reliability, but there is an increase in execution time since the Newton backup is used more often.

In actual runs of CONCHAS the hierarchy performs more reliably than the results of Table III indicate. In fact, with failure of the Gauss–Seidel method signalled after 100 iterations, the hierarchy has never failed during production runs of the code. In addition, there is a synergistic effect in that while previously the Gauss–Seidel method would fail on as many as 40% of the problems, we now find that the backup Newton method is used relatively little. The reason is that previously a single failure would trigger a sequence of failures for a particular cell, but now Newton "fixes" the solution for that cell and eliminates succeeding failures. The result is that for $\Phi > 1$, when the original method tends to fail, the new code runs faster. This improvement is a result of the increased reliability and is not due to any change in the intrinsic speed of the equilibrium solver.

7. CLOSURE

The equilibrium solvers we have considered here are attractive because of their general nature. They are formulated for an arbitrary set of chemical equilibria and can be coded generally to solve different chemical systems.

At least for the Gauss–Seidel method, this "generality" may be illusory. Great care must be taken in choosing the equilibrium reactions for best performance of the method, a procedure which is not systematic. However, the theorem discussed earlier does provide a means to predict the performance for a particular choice of reactions.

Newton's method can be effectively used to back up the Gauss–Seidel method in a hierarchy. In the CONCHAS code, for hydrocarbon combustion, this hierarchy works well, at worst producing only isolated, non-catastrophic failures.

In using the hierarchy, a tradeoff must be made in determining "failure" of the Gauss–Seidel iteration. We tended to use a relatively large number (100) of iterations before invoking Newton for two reasons. First, the Gauss–Seidel method was found always to converge, albeit sometimes very slowly. "Failure" can be regarded as resulting in a less accurate solution and not as a catastrophic error. Second, the Gauss–Seidel result is a better initial guess for Newton than the original Gauss–Seidel start point. If the number of Gauss–Seidel iterations is reduced (say, to 10), Newton is used more often but the "time to failure" of the Gauss–Seidel method is reduced. Proper selection of the maximum number of Gauss–Seidel iterations will minimize total computational time, but this selection is problem-dependent. An additional problem-dependent factor to be considered is that the difference in speed between the Gauss–Seidel and Newton methods depends on the size of the system being solved (i.e., the number of species).

An alternative to the general methods described here is to carry out a preliminary algebraic reduction of the equilibrium equations for the specific problem being considered. We have shown [13] that the hydrocarbon equilibrium system, Eqs. (a–j), can be reduced to a pair of cubic equations, to be solved using Newton's method. The resulting algorithm is intrinsically faster than any of the methods considered here and is 100% reliable. The methodology used to obtain this two-equation system is applicable to other equilibrium problems, so that although the method does require preliminary algebraic manipulation of the equations, it should not necessarily be considered less general.

ACKNOWLEDGMENTS

We thank Roger Krieger and Di Diwakar of GMR for their interest in this work. Valuable discussions were held with Tony Amsden and John Ramshaw of Los Alamos National Laboratory, and with Professor Gary Borman of the University of Wisconsin–Madison.

REFERENCES

1. J. D. RAMSHAW, *Phys. Fluids* **23** (1980), 675.
2. R. D. REITZ AND F. V. BRACCO, *in* "Numerical Methods in Laminar Flame Propagation, A GAMM-Workshop" (N. Peters and J. Warnatz, Eds.), Vieweg-Verlag, Braunschweig, 1982.
3. R. D. REITZ AND F. V. BRACCO, *Combust. Flame* **53** (1983), 141.
4. T. D. BUTLER, L. D. CLOUTMAN, J. K. DUKOWICZ, AND J. D. RAMSHAW, Los Alamos Scientific Laboratories Report LA-8129-MS, 1979.
5. L. D. CLOUTMAN, J. K. DUKOWICZ, J. D. RAMSHAW, AND A. A. AMSDEN, Los Alamos National Laboratory Report LA-9294-MS, 1982.
6. S. GORDON AND B. J. MCBRIDE, NASA SP-273, 1971.
7. C. OLIKARA AND G. L. BORMAN, SAE Paper 750468, 1975.
8. J. D. RAMSHAW AND L. D. CLOUTMAN, *J. Comput. Phys.* **39** (1981), 405.
9. W. ORTEGA AND W. RHEINBOLDT, "Iterative Solution of Nonlinear Equations in Several Variables," Academic Press, New York, 1970.
10. E. ISAACSON AND H. B. KELLER, "Analysis of Numerical Methods," Wiley, New York, 1966.
11. "JANAF Thermochemical Tables," 2nd ed., National Bureau of Standards NSRDS-NBS 37, 1971.
12. "IMSL Library Reference Manual," 8th ed., IMSL Inc., Houston, 1980.
13. K. MEINTJES AND A. P. MORGAN, "Constructing Fast Solvers for Chemical Equilibrium Problems," to be published.