ON THE APPLICATION OF A NEWTON RAPHSON'S ITERATIVE METHOD OF THE FIXED POINT THEORY TO THE SOLUTION OF A CHEMICAL EQUILIBRIUM PROBLEM

Okereke, C. Emmanuel, Ph.D. †

ABSTRACT

In this work, I discussed the solution of a chemical equilibrium problem aiming to obtain its fixed point. To do this, the preliminary and basic ideas introducing the fixed point theory was x-rayed and the Newton Raphson's iterative method for solving the system of non-linear equations discussed, then the problem of the chemical equilibrium involving principal reactions in the production of synthesis gas by partial oxidation of methane with oxygen were stated.

Using a computer program the O reactant ratio that produce an adiabatic equilibrium temperature were obtained by developing a system of seven simultaneous nonlinear equations that has the form which we now solve using the Newton Raphson's method described in section 2.2 and hence the desired fixed point of the chemical equilibrium problem.

THE NETWON'S METHOD, A PRELLIMINARY TO THE NEWTON RAPHSON'S ITERATIVE METHOD

Let T be an operator mapping a set X into itself, a point $x \in X$ is called a fixed point of T if

$$x = T(x) \qquad \ldots \qquad (1.1)$$

By (1.1), we achieve a natural construction of the method of successive approximations.

$$x_{n+1} = T(x_n),\ n \geq 0 \in x \qquad \ldots \qquad (1.2)$$

and if the sequence (x_n), $n \geq 0$ converges to some point $x \, _= \, x^* \in X$ for some initial guess x_0 ε x, where T is a continuous operator in a Banach space X, we have

$$x^* = \lim_{n \to \infty} x_{n+1} = \lim_{n \to \infty} T(x_n) = T\left(\lim_{n \to \infty} x_n\right)$$

that is x* is a fixed point of operations T. Hence we now state without proof the following important results that make easy the understanding of the Newton Raphson's method used in this work

Theorem (1.1): If T is a continuous operator in a Banach X, $\{x_n\}$, $(n \geq 0)$ generated by (1.2) converges to some point $x^* \in X$ for some initial guess $x_0 \in X$ and we say that x* is a fixed point of the operator T [1].

To investigate the uniqueness property, we introduce the concept of contraction mapping as follows. Let $(x, \| \, \|)$ be a metric space and T a mapping of X into itself. The operator T is said to be a contraction if there exists a real number k, $0 \leq k < I$ such that

$$\left\|F(x)-F(y)\right\| \leq k\left\|x-y\right\|, \text{ for all x, y ε x} \qquad \ldots \qquad (1.3)$$

Hence, every contraction mapping T is uniformly continuous. Indeed T is lipschitz continuous with a lipschitz constant k which may also be called the contraction constant for T. With the above; we now discuss the Banach fixed point extensively as related to the target of this research.

Theorem 1.1A [9] (Banach fixed point theorem (1922). Suppose that
we are given an operator $T{:}M \subseteq X \rightarrow M$, i.e., M is mapped into itself by T;
M is a closed nonempty set in a complete metric space (x,d);
T is k – contractive, i.e. $d(T_x, T_y) \leq k$.
Then the following hold:
Existence and uniqueness:- T has exactly one fixed point on M;
Convergence of the iteration: the sequence $\{x_n\}$ of successive approximations converges to the solution, x. for an arbitrary choice of initial point x_0 in M;
Error estimate: for all n = 0, 1, 2… we have the a prior error estimate $d(x_n, x) \leq k^n (1-k)^{-1} d(x_0, x)$, and the a posteriori error estimate $d(x_{n+1}, x) \leq k(1-k)^{-1} d(x_n,x_{n+1})$;
Rate of convergence; for all n =0,1,2,…We have $d(x_{n+1},x) \leq kd(x_n, x)$

Definition 1.1 [9] An operator $T{:}M \subseteq X \rightarrow X$ on a metric space (X,d) is called k- contractive if f (1.3) holds for all x, y ε M with fixed k, $0 \leq k < 1$,
T is called Lipschitz continuous and if.

$$d(Tx,Ty) < d(x,y) \text{ for all x, y ε M with } x \neq y, \qquad \ldots \qquad (1.4)$$

T is called contractive for T and we obviously have the implications:-
k- Contractive → contractive→ no n expansive → Lipschitz continuous.
Every Banach space called the (x,|| ||) also is a complete metric space as

$$(x,d) \text{ under } d(x,y) = \|x\text{-}y\|.$$

On a B-space, (1.3) therefore becomes

$$\|Tx\text{-}Ty\| \leq k\| x\text{-}y\|.$$

On the above the following follows
$\{x_n\}$ is a Cauchy sequence. This follows from

$$d(x_n,x_{n+1}) = d(Tx_{n-1},Tx_n) \leq kd(x_{n-1}.x_n)$$
$$\leq k^2d(x_{n-2},x_{n-1}) \leq \ldots \leq k^nd(x_0,x_1).$$

(1.5)

Repeated application of the triangle inequality and finally summing the formula for a geometric series yields

$$d(x_n,x_{n+m}) \leq d(x_n,x_{n+1}) + d(x_{n+1},x_{n+2})+\ldots+ d(x_{n+m-1}, x_{n+m})$$
$$\leq (k^n +k^{n+1} +\ldots+k^{n+m-1})d(x_0,x_1)$$
$$\leq k^n(1-k)^{-1}d(x_0,x_1)\ldots$$

(1.6)

Since X is complete, the Cauchy sequence converges, i.e., $x_n \rightarrow x$ as $n \rightarrow \infty$ [3].
Equation (1.5) follows by letting $m \rightarrow \infty$.

(II) The error estimate (1.6) follows by letting $m \rightarrow \infty$ in

$$d(x_{n+1},x_{n+m+1}) \leq d(x_{n+1},x_{n+2}) +\ldots+ d(x_{n+m-1}, x_{n+m+1})$$
$$\leq (k +k^2 +\ldots+k^m)d(x_n,x_{n+1})$$

$\leq k(1-k)^{-1}d(x_n,x_{n+m})\ldots$

(1.7)

The point x is a solution of (1.1) for T is continuous by (1.4). Since $T(M) \subseteq M$ and $x_0 \in M$, we have $x_n \in M$ also, for all n. since M is closed and $x_n \rightarrow x$ as $n \rightarrow \infty$, we get $x \in M$. Equation (1.2) implies that $Tx = x$ for $n \rightarrow \infty$.

Equation (1.6) follows $d(x_{n+1,} x) = d(Tx_n,Tx) \leq kd(x_{n,} x)$.

Uniqueness of solution. Suppose $x = Tx$ and $y = Ty$; then $d(x,y) = d(Tx, Ty) \leq kd(x,y)$, which forces $d(x,y) = 0$, i.e. $x = y$.

Continuous Dependence On A Parameter

It is important to note that in many applications, T depends on an additional parameter P. then, (1.1) is replaced by the equation.

$x_p = T_px_p, \quad x_p \in M, \quad \ldots \quad (1.8)$

where $p \in P$.

Proposition 1.2 (Corollary to Theorem 1.1A.) let the following conditions be

P is a metric space, called the parameter space.

For each P, the operator T_p satisfies the hypotheses of Theorem (1.A) but with k in (1.3) independent of p.

For a fixed $p_0 \in P$, and for all $x \in M$, $\lim_{p\rightarrow p0} T_p{}^X = T_{p0}{}^X$.

Then, for each $p \in P$, (1.8) has exactly one solution $x_p \in M$, and $\lim_{p\rightarrow po} x_p = x_{p0}$ [3].

Proof: later x_p be the solution of (1.8) given by theorem 1.1 A, then

$$d(x_p, x_{p0}) = d(T_px_p, T_{p0}x_{p0})$$
$$\leq d(T_px_p,T_px_{po}) + d(T_px_{po}, T_{po}x_{po})$$
$$\leq kd(x_p,x_{po}) + d(T_px_{p0}, T_{p0}x_{p0}),$$

And therefore,

$d(x_p, x_{p0}) < (1-k)^{-1}d(T_px_{po},T_{po}x_{po}) \rightarrow 0$ as $p \rightarrow po$, by (iii). [9]

1.5. Accelerated Convergence and Newton's method [5]

We begin with the insight which underlines the acceleration of iterative methods. Let x be a solution of the real equation $X = F(x)$, and suppose the sequence of iterations (x, where

$x_{n+1} = f(x_n) \quad \ldots$

(1.9)

And x_n €[a, b] for all n, converges to x as $n \rightarrow \infty$.

Now for the key: Suppose further that f is m – times differentiable on [a,b], with

$f'(x) = {}^{f(2)}(x) = \ldots = f^{(m-1)} = 0$

(1.10)

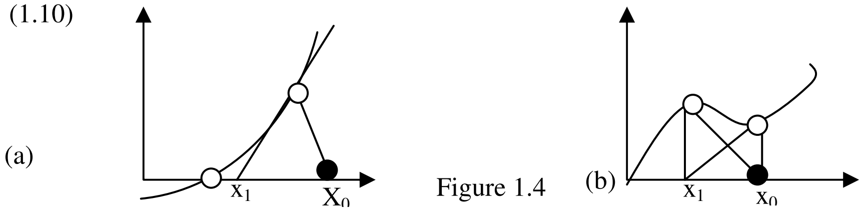


Figure 1.4

Since $x_{n+1} = f(x_n)$ and $x = f(x)$, we have

$|x_{n+1}-x| \leq \sup\limits_{a<\xi<b} f^{(m)}(\xi)|| x_n - x/^m/m!$ - - -

(1.11)

If the supremum in (1.11) is finite, we obtain the convergence of order m, as opposed to the linear convergence (m=1) of (1.9)

Example 1.1[10]. The trick to Newton's method consists of rewriting the equation f(x) = 0 in the equivalent form. X = F(x), where $f(x) = x - \dfrac{f(x)}{f(x)}$

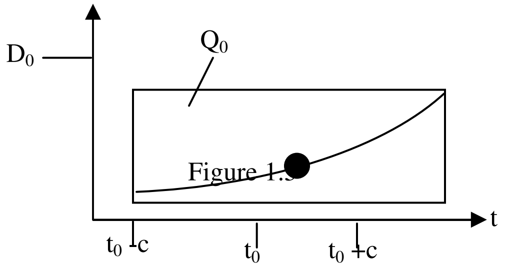
Then the iterative method becomes $x_{n+1} = x_n - \dfrac{f(x_n)}{f(x_n)}$

We assume that $f'(x_n) \neq 0$ for all n. the, $f'(x) = f(x) f^{n}(x) / f'(x)^2$

So that if x is a solution of f (x) = 0 with $f'(x) \neq 0$, then $f'(x) = 0$. Thus we have a method with m = 2 in (1.1.9) i.e. we have quadratic convergence.

We apply this to the equation x = T(x) in (1.1.1) i.e. f(x) = T(x). Computing, we obtain the iterative values x_n with linear convergences.

The geometric interpretation of Newton's methods is seen in figure 1.4 (a). To find a zero, x, of f, take the initial value, x_0, and determine the corresponding functional value, $f(x_0)$. The next iterative value, x_1 is the intersection of the tangent line at $(x_0,f(x_0))$ and the x- axis. Keep repeating the process, it is typical of Newton's method that it converges very rapidly if the initial value x_0 is already in the vicinity of the zero, but figure 1.4(b) shows a better of this.



However, we know that the above discussed fixed point method is just the traditional fixed point method that is restricticted to the solution of only linear systems and for the purpose of this research we advance onto the modified Newton's method which is the Newton Raphson's iterative method here below generated for use in section three

2. NEWTON'S RAPHSON'S METHOD

Sections 2 and 3 are concerned with finding the solution, or solutions, of the system.

$$
\begin{aligned}
&f_1(x_1,x_2,\ldots,x_n)=0,\\
&f_2(x_1,x_2,\ldots,x_n)=0,\ \vdots \qquad\qquad (2.1)\\
&f_n(x_1,x_2,\ldots,x_n)=0,
\end{aligned}
$$

Involving n real functions of the n real variables $x_1,x_2,\ldots,x_n$. Following the previous notation, $x = \{x_1,x_{2,\ldots,}x_n\}^t$, we shall write $f_i\ (x) = f_i\ (x_1,x_2,\ldots,x_{n)}$ here, and in the subsequent development, $1 \leq i \leq n$. Then let $\alpha = [\alpha_{1,}\alpha_2,\ldots,\ \alpha_n]^t$ be a solution of (2.1), that is, let $fi(\alpha) = 0$.

Let the n functions f_i (x) be such that $x_i = F_i\ (x)$ (2.2)

Implies $f_i\ (x) = 0,\ 1 \leq j \leq n$. Basically, the n equations (2.2) will constitute a suitable rearrangement of the original system (2.1). In particular,

let $\alpha_i = f_i\ (\alpha)$ … (2.3)

Let the starting vector $x_0 = [x_{10},\ x_{20},\ldots,x_{n0}]^t$ be an approximation to α. Define successive new estimates of the solution vector, $x_k = [x_{1k},x_{2k}\ldots,x_{nk}]^t$, $k = 1, 2,\ldots,$ by computing the individual elements from the recursion relations.

$x_{ik} = F_i\ (x_{1,k-1,}\ x_{2,k-1,\ldots,}x_{n,k-1}).$ (2.4)

suppose there is starting R describable as $|x_j - \alpha_j| \leq h,\ 1 \leq j \leq n$, and for x in R there is a positive number μ, less than one, such that

$$\sum_{j=1}^{n}\left|\frac{\partial F_i(x)}{\partial x_j}\right| \leq \mu \qquad (2.5)$$

Then, if the starting vector x_0 lies in R, we show that the iterative method expressed by (2.4)

converges to a solution of the system (2.1), that is,

$$\lim_{k\to\infty} x_k = \alpha \qquad (2.6)$$

Using the mean-value theorem, the truth of (2.1) is established by first noting from (2.3) and (2.4), that

$x_{ik}- \alpha_I = F_i\ (x_{k-1})\text{-}F_i\ (\alpha).$

$$= \sum_{j=1}^{n}(x_{j,k-1} - \alpha_j)\frac{\partial F_i\left[\alpha + \xi_{i,k-1}(x_{k-1} - \alpha)\right]}{\partial x_j}, \qquad (2.7)$$

In which $0 < \xi_{i,k-1} < 1$. that is,

$$|x_{ik} - \alpha| \leq h\sum_{j=1}^{n}\left|\frac{\partial F_i}{\partial x_j}\right| \leq \mu h < h, \qquad (2.8)$$

Showing that the points x_k lie in R. also, by induction, from (2.5) and (2.7),

$$|x_{ik} - \alpha| \leq \mu \max (|x_{j,k-1} - \alpha_j|) \leq \mu^k h. \qquad (2.9)$$

Therefore, (2.6) is true, and the procedure converges to a solution of (2.1). Note that if the F_i (x) are linear, we have the Newton's method, and the sufficient conditions of (2.5) are the same as the second set of sufficient conditions controlling the Newton's iteration.

For the nonlinear equations, there is also a counterpart to the Newton's method, previously discussed for the linear case. We proceed as before, except that some replacements are made by

$$X_{ik}=F_i\,(x_{ik},x_{2k},\ldots,x_{i-1,k},x_{i,k-1},\ldots x_{n,k-1}). \qquad (2.10)$$

That is, the most recently computed elements of the solution vector are always used in evaluating the F_i. The proof of convergence according to (2.10) is much the same as for the jacobi-type iteration. We have

$$x_{ik} - \alpha_i = \sum_{j=1}^{n} (x_{j,k-1} - \alpha_j) \frac{\partial F_i(\Sigma_{ik})}{\partial x_j},$$

Where $\sum X_{ik} - \alpha = [\alpha_1 + \xi_{ik}(x_{1,k-1} - \alpha),\ldots,\alpha_n + \xi_{ik}(x_{n,k-1} - \alpha_n]^t$,

It will appear inductively that the above is true, because the various points concerned remain in R. If e_{k-1} is the largest of the numbers $|x_{j,k-1} - \alpha_j|$, then

$$x_{ik} - \alpha_1| \leq \mu e_{k-1} < e_{k-1} < h.$$

it follows that

$$x_{2k} - \alpha_2 = (x_{ik} - \alpha_1)\frac{\partial F_2(\Sigma_{2k})}{\partial x_j} + \sum_{j=2}^{n}\left(x_{j,k-1} - \alpha_j\right)\frac{\partial F_2(\Sigma_{2k})}{\partial xj}$$

where

$\varepsilon_{2k} = [\alpha_1 + \xi_{2k}(x_{1k}-\alpha_1),\ \alpha_2+\xi_{2k}(x_{2,k-1}-\alpha_2),\ldots\ \alpha_n + \xi_{2k}(x_{n,k-1} - \alpha_n)]^t$. that is, $|x_{2k}-\alpha_2| \leq \mu e_{k-1} < e_{k-1} < h$.

Therefore, $|x_{ik} - \alpha_i| \leq \mu^k h$, and convergence according to (2.1) is again established.

Observe that the first of the sufficiency conditions of the same (2.10) has been reaffirmed under slightly general circumstance.

2.2 Newton- Raphson's Iteration for Nonlinear Equations.

The equations to be solved are again those of (2.1), and we retain the nomenclature of the previous section. The Newton-Raphson process, to be described, is once more iterative in character. We first define.

$$f_{ij}(x) = \frac{\partial f_i(x)}{\partial x_j} \qquad \ldots \qquad (2.11)$$

Next define the matrix ϕ (x) as

$$\phi(x) = (f_i(x)),\ 1 \leq I \leq n,\ 1 \leq j \leq n. \qquad \ldots \qquad (2.12)$$

Thus det (ϕ (x)) is the *Jacobian* of the system (2.1) for the vector $x = [x_1, x_2,\ldots,x_n)^t$. now define the vector

$$f(x) \text{ as } f(x) = [f_i(x), f_2(x),\ldots, f_n(x)]^t. \qquad (2.13)$$

With these definitions in mind, and with the starting vector

$$x_0 = [x_{10}, x_{20}, \ldots, x_{n0}]^t, \text{ let } x_{k+1} = x_k + \delta_k, \quad \ldots \quad (2.14)$$

The fundamental theorem concerning convergence is much less restrictive than those of the previous sections. We have the result that if the components of ϕ (x) are continuous in a neighborhood of a point α such that

$f(\alpha) = 0$, if det $(\phi(\alpha)) \neq 0$, and if x_0 is "near" α, then $\lim_{k\to\infty} x_k = \alpha$.

… (2.15)

An outline for a method of proof follows. By (2.13) and (2.14), since $f_i(\alpha) = 0$, $\delta_k = \phi^{-1}(x_k)\,[f(\alpha)]$. …

(2.15)

By the mean- value theorem,

$$f_i(x_k) - f_i(\alpha) = \sum_{j=1}^{n} \int ij(\alpha + \xi_{ik}(x_k - \alpha))(x_{jk} - \alpha),$$

$$where \quad 0 < \xi_{ik} < 1.$$

For the ith row of a matrix ψ use $[f_{11}(\alpha + \xi_{ik}(x_k - \alpha)), \ldots, f_{in}(\alpha + \xi_{ik}(x_{k-}\alpha))]$.

Then $x_{k+1} - \alpha = x_k - \alpha + \delta_{k=}\phi^{-1}(x_k)\,\phi(x_k) - \psi]\,(x_k - \alpha)$.

Since the entries in the matrix $\phi(x_k) - \psi$ are differences of the type $f_{ij}{}_{(}x_k) - f_{ij}(\alpha + \xi_{ik}(x_k - \alpha))$, they can be kept uniformly small if the starting vector X_0 lies in an initially chosen region R describable as $|x_i - \alpha_i| \leq h$, $1 \leq i \leq n$. concurrent with this is the fact that since det $(\phi(x_k))$ can be bounded from zero. The net result is that, for $0 < \mu < 1$, $|x_{ik} - \alpha_i| \leq h\,\mu^k, 1, \leq i \leq n$. thus the sequence $[x_k]$ converges to α [6].

Example (2.1) *[7]*. To illustrate the procedure, we use the example below namely.

$$f_1(x_1, x_2) = \frac{1}{2}\sin(x_1, x_2) - \frac{x_2}{4\pi} - \frac{x_1}{2} = 0$$

$$f_2(x_1, x_2) = \left(1 - \frac{1}{4\pi}\right)(e^{2x1} - e) + \frac{ex_2}{\pi} - 2ex1 = 0. \quad \ldots \quad (2.16)$$

it is readily seen that

$$\frac{\partial f_1}{\partial x_1} = -\frac{1}{2}\frac{x_2\cos(x_1, x_2)}{2}, \quad \frac{\partial f_1}{\partial x_1} = -\frac{1}{4\pi} + \frac{x_1\cos(x_1, x_2)}{2},$$

$$\frac{\partial f_2}{\partial x_1} = -2e + \left(2 - \frac{1}{2\pi}\right)e^{2x1}, \quad \frac{\partial f_2}{\partial x_2} = \frac{e}{\pi},$$

The increments Δx_1 and Δx_2 in x_1 and x_2 are determined by

$$\frac{\partial f_1}{\partial x_1}\Delta x_1 + \frac{\partial f_1}{\partial x_2}\Delta x_2 = -f_1,$$

$$\frac{\partial f_2}{\partial x_1}\Delta x_1 + \frac{\partial f_2}{\partial x_2}\Delta x_2 = -f_2.$$

Or, writing the determinant D of the coefficient matrix (the Jacobian),

$$D = \frac{\partial f_1}{\partial x_1}\frac{\partial f_2}{\partial x_2} - \frac{\partial f_1}{\partial x_2}\frac{\partial f_2}{\partial x_1},$$

then

$$\Delta x_1 \left(\frac{f_2 \dfrac{\partial f_1}{\partial x_2} - f_1 \dfrac{\partial f_2}{\partial x_2}}{D} \right), \Delta x_2 \left(\frac{f_1 \dfrac{\partial f_2}{\partial x_1} - f_1 \dfrac{\partial f_1}{\partial x_1}}{D} \right).$$

for case in verification, detailed results are tabulated in Table 3.1 and moreover, calculations were carried out using slide rule and the entries 0.00000 showed tiny negative values.

APPLICATION OF NEWTON- RAPHSON'S METHOD IN SOLVING THE CHEMICAL EQUILIBRUIM PROBLEM

The principal reactions in the production of synthesis gas by partial oxidation of methane with oxygen are:

$CH_4 + ½\, O_2$ → $CO + 2H_2$ → - - - (3.1)

$CH_4 + H_2O$ → $CO + 3H_2$ → - - - (3.2)

$H_2 + CO_2$ → $C0 + H_2O$ → - - - (3.3)

Write a program that finds the 0 reactant ratio that will produce an adiabatic equilibrium temperature of 2200^0 F at an operating pressure of 20 atmospheres, when the reactant gases are preheated to an entering temperature of 1000^0F.

Assuming that the gases behave ideally, so that the component activities are identical with component partial pressures, the equilibrium constants at 2200^0F for the three equations are respectively:

$$K_1 = \frac{PcoP^2{}_{H2}}{P_{CH}P^{1/2}_{02}} = 1.3x10^{11} \downarrow \quad \rightarrow \quad \dots \qquad (3.4)$$

$$K_2 = \frac{PcoP^3{}_{H2}}{P_{CH\,4}P_{H\,2O}} = 1.7837x10^5 \rightarrow \quad \dots \qquad (3.5)$$

$$K_3 = \frac{P_{CO}P_{H_20}}{P_{CO_2}P_{H_2}} = 2.6058 \quad \rightarrow \qquad (3.6)$$

Here P_{CO}, Pco_2, $P_{H2O,}$ P_{CH4} and P_{O2} are the partial pressured of CO (carbon monoxide), CO_2 (carbon dioxide), H_2O (water vapor), H_2 (hydrogen), CH_4 (methane), and O_2 (oxygen), respectively. Enthalpies of the various components at 1000^0F and 2200^0F are listed in Table (3.1)

Table (3.1) Component Enthalpies in BTU/b mole

Component	1000^0F	2200^0F
CH_4	-13492	8427
H_2O	-90546	-78213
CO_2	-154958	-139009
CO	-38528	-28837
H_2	10100	18927
O_2	10690	20831

A fourth reaction may also occur at high temperatures:
C+ CO_2 2CO (3.1) at 2200^0F, any carbon formed would be deposited as a solid; the equilibrium constant is given by

$$K_4 = \frac{P^2 co}{ac \quad Pco_2} = 1.7837 \quad x10^5 \qquad (3.8)$$

where a_c is the activity of carbon in the solid state. Do not include reaction (3.7) in the equilibrium analysis. After establishing the equilibrium composition, considering only the homogeneous gaseous reactions given by (3.1), (3.2), and (3.3), determine the thermodynamic likelihood that solid carbon would appear as a result of reaction (3.7). Assume that the activity of solid carbon is unaffected by pressure and equals unity.

Use the Newton- Raphson method to solve the system of simultaneous nonlinear equations developed as the result of the equilibrium analysis.

3.2 METHOD OF SOLUTION

Because of the magnitude of K, the equilibrium constant for reactions, the first reaction can be assumed to go to completion at 2200^0F, that is virtually no unrelated oxygen will remain in the product gases at equilibrium.

Let the following nomenclature be used.

x_1 mole fraction of CO in the equilibrium mixture
x_2 mole fraction of CO_2 in the equilibrium mixture
x_3 mole fraction of H_2O in the equilibrium mixture
x_4 mole fraction of H_2 in the equilibrium mixture
x_5 mole fraction of CH_4 in the equilibrium mixture
x_6 number of moles of O_2 per mole of CH_2 in the feed gas
x_7 number of moles of product gases in the equilibrium mixture per mole of CH_4 in the feed gases.

Then a system of seven simultaneous equations may be generated from three atom balances an energy balance, a mole fraction constraint and two equilibrium relations.

Atom conservation balances: the number of atoms of each element entering equals the number of atoms of each clement in the equilibrium mixture.

Oxygen: $x_6 = (1/2x_1 + x_2 + 1/2x_3$ → … (3.9)
Hydrogen: $4 = (2x_3 + 2x_4 + 4x_2$ → … (3.10)
Carbon: $1 = (x_1 + x_2 + x_5$ → … (3.11)

Since the reaction is to be conducted adiabatically, that is, no energy is added to or removed from the reacting gases, the enthalpy (H) of the reactants must equal the enthalpy of the products.

$[HCH_4 + x_6H0_2]_{1000'F} = x_7[x_1HC0_2 + x_3H_20 \quad + x_4H_2 + x_5HCH_2]_{2200'F}$ … (3.12)

mole fraction constraint.

$$x_1 + x_2 + x_4 + x_5 + = 1 \qquad \ldots \qquad (3.13)$$

Equilibrium relations.

$$K_2 = \frac{p^2 X_1 X^3{}_4}{x_3 x_5} = 1.7837 \ x10^5 \rightarrow \qquad \ldots \qquad (3.14)$$

$$K_3 = \frac{X_1 X_3}{X_2 X_4} = 2.6058 \rightarrow \qquad \ldots \qquad (3.15)$$

The relationships (3.14) and (3.15) follow directly from (3.5) and (3.6), respectively, where P is the total pressure and $P_{co} = Px_{1,}$ etc. in addition, there are five side conditions.

$$x_1 \geq 0, I = 1, 2, 5 \quad \ldots \qquad (3.16)$$

These C ions more that all mole fractions in the equilibrium mixture are nonnegative, that is, any solution of equation (3.9) to 3.15) that contains negative mole fractions is physically meaningless from physical-chemical principle there is one and only one solution of the equation that satisfies conditions (3.16). Any irrelevant solutions may be detected easily.

The seven equations may be rewritten in the form.

$f_1\ (x) = 0, \qquad I = 1, 2,\ldots, 7 \qquad (5.5.17)$

where $x = [x_1x_2\ x_3\ x_4\ x_5\ x_6\ x_7]^t$, as follows:

$$f_1(x) = \frac{1}{2}x_1 + x_2 + \frac{1}{2}x_3 - \frac{x_6}{x_7} = 0 \rightarrow \qquad (3.19\ a)$$

$$f_2(x) = x_3 + x_4 + 2x_5 - \frac{2}{x_7} = 0 \rightarrow \qquad (3.19\ b)$$

$$f_3(x) = x_1 + x_2 + x_5 - \frac{1}{x_7} = 0 \rightarrow \qquad (3.19\ c)$$

$$f_4(x) = -28837 \ x_1 - 139009 \ x_2 - 78213 \ x_3 + 18927 \ x_4 + 8427 \ x_5 + \frac{13492}{x_7} - 10690 \ \frac{x_6}{x_7} = 0 \rightarrow \qquad (3.19\ d)$$

$$f_5(x) = x_1x_2x_3x_4x_5 - 1 = 0 \rightarrow \qquad (3.19\ e)$$

$$f_6(x) = P^2x_1x_4 - 1.7837 \ x10^5 x_3x_5 = 0 \rightarrow \qquad (3.19\ f)$$

$$f_7(x) = x_1x_3 - 2.6058 \ x_2x_4 = 0 \rightarrow \qquad (3.19\ g)$$

The system of simultaneous nonlinear equations has the form (2.1), and will be solved using the Newton- Raphson method, described in section 2.2. The partial derivatives of above may be found by partial differentiation of the seven functions, f_1 (x), with respect to each of the seven variables. For example,

$$\frac{\partial f_1}{\partial x_1} = \frac{1}{2}, \qquad \frac{\partial f_1}{\partial x_4} = 0, \qquad \frac{\partial f_1}{\partial x_7} = \frac{x_6}{x_7^2}$$

$$\frac{\partial f_1}{\partial x_{21}} = 1, \qquad \frac{\partial f_1}{\partial x_5} = 0,$$

$$\frac{\partial f_1}{\partial x_3} = \frac{1}{2}, \qquad \frac{\partial f_1}{\partial x_6} = -\frac{1}{x_7},$$

The Newton- Raphson method may be summarized as follows:
Choose a starting vector $x_K = x_0 = [x_{10}, x_{20}, \ldots, x_{70}]$, where x_0 is hopefully near a solution
solve the system of linear equations (2.14),
$\phi\ (x_k)\delta_k = -\ f(x_k)$,
where

$$\phi_{ij}\ (x_k) = \frac{\partial f_1}{\partial x_j}(x_k) \qquad \begin{matrix} i = 1, 2, \ldots, & 7, \\ j = 1, 2, \ldots, & 7, \end{matrix} \qquad (3.20)$$

$$f(x_k) = lf_1(x_k),\ f_2(x_k), \ldots\ldots, \quad f_7(x_k))^t, \qquad (2.21)$$

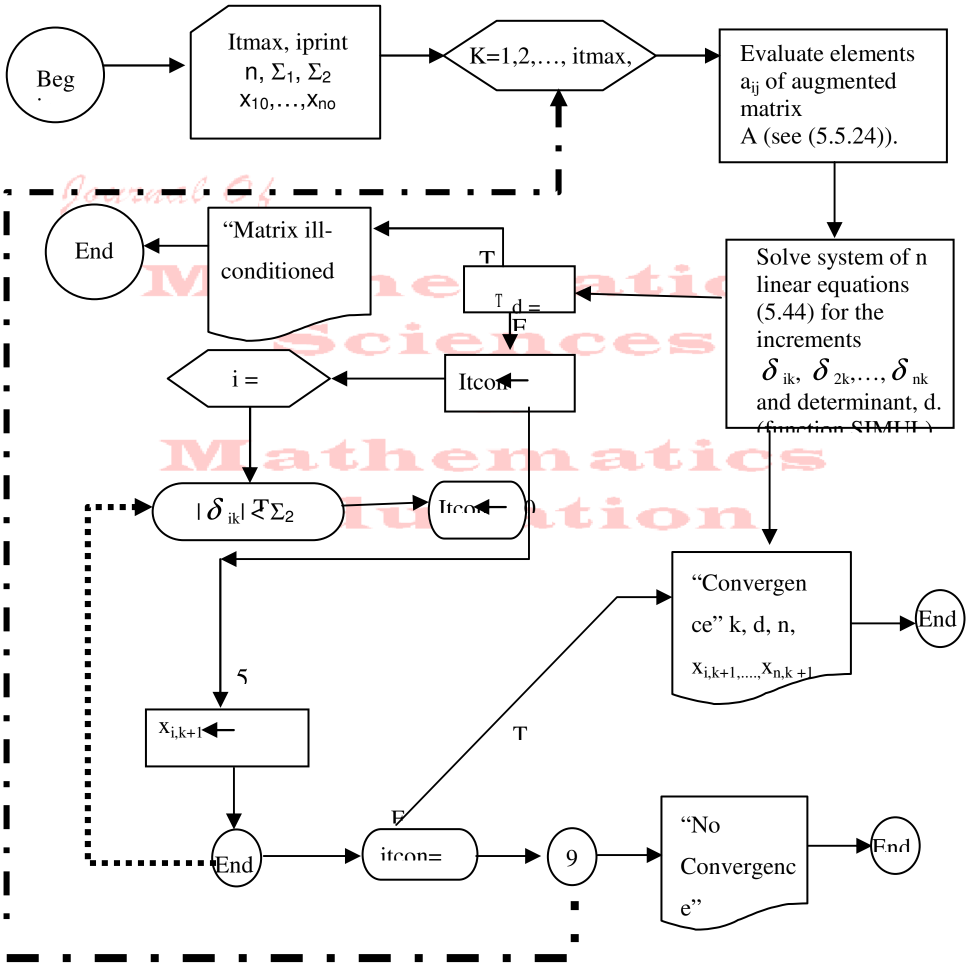
for the increment vector

$$\delta_k = \left[\delta_{1k}\ \delta_{2k}, \ldots\ldots, \quad \delta_{7k} \right]^t \qquad (3.22)$$

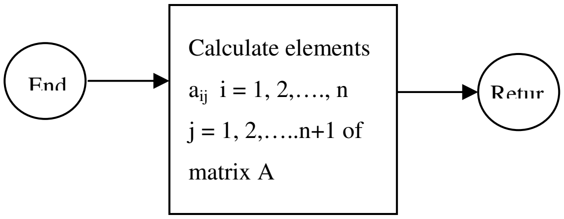
update the approximation to the root for the next iteration. $x_{k+1} = x_k + \delta_k$.
check for possible convergence to a root α. One such test might be
$|\delta_{ik}| < \varepsilon_2$, $\quad i = 1,2,\ldots,7.$ … (3.23)
if (3.23) is true for all i, then $x_k +_1$ is taken to be the root. Is test (3.23) is failed for any 1, then the process is repeated starting with step 2. The iterative process is continued until test (3.23) is passed for some k, or when k exceeds some specified upper limit. In the programs that follow, the elements of the augments matrix $A=[\phi(x_k):\ -\ f(x_k)]$ … (3.24) Are evaluated by a subroutine named CALCN. The system of linear equations (3.24) is solved by calling on the function SIMUL, described in detail in *example* (2.1)
The main program is a general one, in that it is not specifically written to solve only the seven equations of interest. By properly defining the subroutine CALCN, the main program could be used to solve any system of n simultaneous nonlinear equations. The main program reads data values for itmax, iprint, n, Σ_1, Σ_2, and $x_1, x_2, \ldots, x_n$ here, itmax is the maximum number of Newton- Raphson's iterations, print is a variable that controls printing of intermediate output, n is the number of nonlinear equations, Σ_1, is the minimum pivot magnitude allowed in the Gauss-Jordan reduction algorithm, Σ_2, is a small positive number used in test (3.23), and $x_1, x_{20}, \ldots, x_{no}$, that is, the elements of x_0.

FLOW DIAGRAM

Main Program



Subroutine CALC (Arguments: x_k, A, N)



Fortran Implementation

List of Principle various program Symbol

(Main)

A Augmented matrix of coefficients, A (see (3.22).

DETER d, determinant of the matrix ϕ (the jacobian).

EPS1 Σ_1, minimum pivot magnitude permitted in subroutine SIMUL.

EPS2 Σ_2, small positive number, used in convergence test (3.23). subscript, *i.*

IPRINT Print control variable, if iprint = 1, intermediate solutions are printed after each iteration.

ITCON used in convergence test (3.23). ITCON 1 if (3.23) is passed for all i, i = 1, 2,…..,n: otherwise ITCON = 0.

ITER Iteration counter, k.

ITMAX maximum number of iterations permitted, itmax.

N number of nonlinear equations, n.

XINC vector of increments, $_{ik}$, i = 1, 2, ….,n.

XOLD vector of approximations to the solution, x_{ik}.

SIMUL function developed in Example (2.1) solves the system of n linear equations (2.15) for the increments,. $_{ik}$ i = 1, 2,…,n.

(subroutine CALCN)

DXOLD same as XOLD. Used to avoid an excessive number of reference to subroutine arguments in CALCN.

I, J, i and j, row and column subscript, respectively.

NRC N, dimension of the matrix A in the calling program. A is assumed to have the same number of rows and columns.

P pressure, P, atm.

Program Listing*(Chemical Equilibrium by Newton- Raphson Method)*

Main Program

```
c        APPLIED NUMERICAL METHODS, EXAMPLES 5.5
C        CHEMICAL EQUILIBRIUM- NEWTON-RAPSION METHOD
C
C        THIS PROGRAM SOLVES N SIMULTANEOUS NON-LINEAR
EQUATIONS
C        IN N UNKNOWNS BY THE NEWTON-RAPSON ITERATIVE
PROCEDURE
C        INITIAL GUESSES FOR VALUES OF THE UNKNOWNS ARE
READ INTO
C        XOLD (1)…….XOLD (N). THE PROGRAM FIRST CALLS ON
THE SUBROUTINE
C        CALCN TO COMPUTE THE ELEMENTS OF A. THE
AUGMENTED MATRIX OF
```

```
C              PARTIAL DERIVATIVES, THEN ON FINCTION SIMUL (SEE
PROBLEM 5.2)
C             TO SOLVE THE GENERATED SET OF LINEAR EQUATIONS
FOR THE CHANGES
C              IN THE SOLUTION VALUES XINC(1)……XINC(N). DETER
IS THE
C             JOCABIAN COMPUTED BY SIMUL. THE SOLUTIONS ARE
UPDATED AND THE
C             PROCESS CONTINUED UNTIL ITER, THE NUMBER OF
ITERATIONS,
C             EXCEEDS ITMAX OR UNTIL THE CHANGE IN EACH OF
THE N VARIABLES
C             IS SMALLER IN MAGNITUDE THAN EPS2 (ITCON =1
UNDER THESE
C             CONDITIONS). EPS1 IS THE MINIMUM PIVOT MAGNITUDE
PERMITTED
C             IN SIMUL. WHEN IPRINT  =1, INTERMEDIATE SOLUTION
VALUES ARE
C             PRINTED AFTER EACH ITERATION.
C
          DIMENSION XOLD (21), XINC (21), A (21,21)
C
C         ….READ AND PRINT DATA…
    1    READ (5,100) ITMAX, IPRINT, EPS1, EPS2, (XOLD(1), 1=1, N)
          WRITE(6,200) ITMAX, IPRINT, N, EPS1, EPS2, N, (XOLD)(1),
1=1,N)
C
C        … CALL ON CALCN TO SET UP THE A MATRIX…
        CALL CALCN (XOLD, A, 21)
C
C        … CALL SIMUL TO COMPUTE JACOBIAN AND CORRECTIONS
IN XINC…
        DETER = SIMUL (N, A, XINC, EPS1, 1, 21)
        IF (   DETER. NE. D. ) GO TO 3
            WRITE (6   GO TO

C
C         …CHECK FOR CONVERGENCE AND UPDATE XOLD VALUES…
    3    ITCON = 1
         DO 5 1 – 1, N
         IF (ABS(XINC(1)) . GT. EPS 2 )  ITCON = 0
   5        XOLD (1) = XOLD(1) + XINC(1)
       I F   (   IPRINT. EQ. 1   )     WRITE  (6, 202)   ITER, DETER, N, (XOLD (1), 1 =
1,N)
       I F   (   ITCON. EQ. 0  )    ITER, N, (XOLD(1), 1 = 1, N)
           WRITE (6,203) ITER, N, (XOLD(1), 1 =1,N)
           GO TO 1
   9     CONTINUE
C
          WRITE 96, 204)
```

PROGRAM Listing (continued)

Subroutine CALCN

```
        SUBROUTINE CALCN D XOLD, A, NRC  )
C              THIS SUBROUTINE SETS UP THE AUGMENTED MATRIX OF
PARTIAL
C              DERIVATIVES REQUIRED FOR THE SOLUTION OF THE
NON-LINEAR
C              EQUATIONS WHICH DESCRIBE THE EQUILIBRIUM
CONCENTRATIONS
C              OF CHEMICAL CONSTITUENTS RESULTS FROM PARTIAL
OXIDATION
C              OF METHANE WITH OXYGEN     TO PRODUCE SYNTHESIS
GAS.     THE PRESSURE
C              IS 20 ATMOSPHERES.     SEE     TEXT     FOR MENINGS OF
XOLD(1)…XOLD(N)
C              AND A LISTING OF THE EQUATIONS.       DXOLD HAS BEEN
USED AS THE
C              DUMMY    ARGUMENT  FOR    XOLD  TO AVOID   AN
EXCESSIVE NUMBER OF
C              REFENCENCES TO ELEMENTS IN THE ARGUMENT LIST.
C
       DIMENSION XOLD(20), DXOLD(NRC), A(NRC , NRC)
C
       DATA P /   20.  /
C
C         …SHIFT ELEMENTS OF DXOLD TO XOLD AND CLEAR A
ARRAY…
       DO       1    I  = 1 ,     7
       XOLD(1)       =       DXOLD(1)
       DO      1       J = 1,     8
 1   A(1 , J)  = 0.
C
C          …COMPUTE NON-ZERO ELEMENTS OF A…
           A (1,1) = 0.5
           A (1,2) = 1.0
           A (1,3) = 0.9
           A (1,6) =  1.0 / XOLD (7)
           A (1,7) = XOLD (6) / XOLD (7)**2
           A (1,8) = - XOLD(1) / 2. – XOLD (2) -  XOLD (3) /2,   * XOLD (6) /
XOLD(7)
           A (2,3) = 1.0
           A (2,4) = 1.0
           A (2,5) = 2.0
           A (2,7) = 2.0 / XOLD (7) **2
           A (2,8) = XOLD(1) – XOLD(2) – XOLD(5) + 1.0 / XOLD(7
      A (3,1) = 1.0
      A (3,2) = 1.0
      A (3,5) = 1.0
      A (3,7) = 1.0 / XOLD(7) **2
      A (3,8) = - XOLD(1) – XOLD(2) – XOLD(5) + 1.0 / XOLD(7)
      A (4,1) = - 28837.
      A (4,2) = - 139009.
      A (4,3) = - 78213.
      A (4,4) = 18927
      A (4,5) = 8427
      A (4,6) = - 10690. / XOLD (7)
      A (4,7) = (- 13492. + 10690. * XOLD (6) ) / XOLD (7)**2
      A (4,8) = 28837. * XOLD(4) – 8427. * XOLD(5) – 13492. / XOLD 97) + 10690.
     1             -18927. * XOLD(4)  - 8427. * XOLD(5) – 13492. / XOLD 97)  + 10690.
     2             * XOLD(6) / XOLD(7)
      A (5,1) = 1.0
      A (5,2) = 1.0
      A (5,3) = 1.0
      A (5,4) = 1.0
      A (5,5) = 1.0
      A (5,8) = 1.0 – XOLD(1) – XOLD(2)  -  XOLD(3)  - XOLD(4)  -  XOLD(5)
      A (6,1) = P*P* XOLD(4)**3
      A (6,3) = - 1.7837E5 * XOLD (5)
      A (6,4) = 3.0 *P*P* XOLD(1) * XOLD(4)**2
      A (6,5) = - 1.7837E5* XOLD (3)
      A (6,8) = 1.7837E5*XOLD(5)*XOLD(3) – P*P*XOLD(1)*XOLD(4)**3
      A (7,1) = XOLD (3)
      A (7,2) = - 2.6058*XOLD(4)
      A (7,3) = XOLD(1)
      A (7,4) = - 2.6058* XOLD(2)
      A (7,8) = 2.6058* XOLD(4) *XOLD(2) -  XOLD(1)*XOLD(3)
      RETURN
C
      END
```

Chemical Equilibrium (Newton- Raphson Method)

Program listing (continued)

Data

```
   ITMAX   =   50                IPRINT    = 1             N          =
7
   EPS1     = 1.0E – 10           EPS2      = 1. 0E – 05
   XOLD (1)… XOLD(5)      =           0.500          0.000          0.000
0.500          0.000
   XOLD(6)….XOLD(7)      =           0.500          2.000
    ITMAX    =       50              IPRINT    =     0             N
=   7
    EPS1      =     1.0E-10           EPS2      =     1.0E – 05
    XOLD(1)….XOLD(5)     =           0.200          0.200          0.200
0.200          0.200
    XOLD(6)….XOLD(7)     =           0.500          2.000
    ITMAX    =       50              IPRINT    =     0             N
=    7
```

```
    EPS1      =     1.0E  - 10           EPS2      =      1.0E -05
    XOLD(1)….XOLD(5)       =      0.220                      0.075
0.001        0.580          0.125
    XOLD(6)….XOLD(7)       =      0.436                      2.350
```

Computer Output
Results for the 1st Data Set

```
    ITMAX             =             50
    IPRINT          =               1
    N               =              7
    EPS1            =             1.0E -10
    EPS2            =             1. 0E -05
                              XOLD(1)....XOLD (     7)
          5.000000E -01                 0.0                       0.0
5.000000E-01
          0.0                           5.000000E -01            2.000000E   00
    ITER             =             1
    DETER            =            -0.97077E         07
                             XOLD(1)...XOLD(    7)
         2.210175E -01                  2.592762E -02             6.756210E -02
4.263276E -01
         2.591652E -01                  3.3432350E -01            1.975559E   00
    ITER            =              2
    DETER           =              -0.10221E        10
                                XOLD(1)…XOLD(      7)
         3.101482E -01                  7.142063E -03             5.538273E -02
5.791981E-01
         4.812878E -02                  4.681466E -01             2.524948E  00
   ITER             =              3
   DETER            =              -0.41151E        09
                             XOLD(1)…XOLD(    7)
        3.202849E -01                   9.554777E -03             4.671279E -02
6.129664E -01
        1.048106E-02                    5.533223E -01             2.880228E   00
  ITER             =               4
  DETER            =               -0.22807E        09
                            XOLD(1)…XOLD(      7)
        3.228380E -01                   9.22480E -03              4.603060E -02
6.180951E-01
        3.811378E -03                   5.758237E -01             2.974139E   00
  ITER              =              5
  DETER             =              -0.20218E       09
                            XOLD(1)…XOLD(     7)
        3.228708E -01                  9.223551E -03              4.601710E -02
6.181716E -01
```

```
          3.716873E -03             5.767141E -01            2.977859E  00
   ITER              =               6
   DETER             =              -0.20134E      09
                           XOLD(1)…XOLD(      7)
          3.228708E -01             9.223547E -03            4.601710E -02
6.181716E -01
          3.716847E -03              5.767153E -01           2.977863E  00
```

Computer Output

```
SUCCESSFUL CONVERGENCE
     ITER              =             6
                                                XOLD(1)…XOLD(    7)
              3.228708E -01                   9.223547E -03
4.601710E -02                   6.181716E -01
              3.716847E -03                   5.767153E-01            2.97863E
00
```

Results for the 3rd Data Set

```
    ITMAX              =             50
    IPRINT             =             1
    N                  =             7
    EPS1               =             1.0E -10
    EPS2               =             1. 0E -05
                                 XOLD(1)...XOLD (      7)
            2.200000E -01                      7.499999e -02
9.999999e -04                    5.800000E-01
            1.250000e -01                     4.360000e -01             2.349999e
00
    ITER               =            1
    DETER              =           -0.61808E  08
                                XOLD(1)...XOLD(    7)
           6.9514955E -01                    -8.022028E -02             1.272939E
-02                  1.217132E  00
          -8.447912E  -01                     1.314754E  00             5.969404E
00
                      ITER                 =           2
    DETER             =             0.12576E  09
                                XOLD(1)…XOLD(     7)
          4.958702E -01                      -1.698154E -02             5.952045E
-03                  9.518250E -01
         -3.65007E -01                        2.379797E  00
1.043425E  01
    ITER               =            3
    DETER             =             0.77199E  07
                                     XOLD(1)…XOLD(    7)
         4.559822E -01                          -9.799302E -04            -7.583648E -04
9.107630E -01
        -3.650070E -01                           2.509821E  00             1.107038E  01
   ITER            =              4
   DETER           =              0.53378   07
                              XOLD(1)…XOLD(      7)
        4.569673E -01                          -4.071472E -04            -2.142648E -03
9.152630E -01
       -3.696806E -01                           2.608933E 00              1.149338E  01

   ITER            =              5
   DETER           =              0.49739E  07
                              XOLD(1)… XOLD(     7)
       4.569306E -01                          -4.071994E -04             -2.125205E -03
9.151721E -01
      -3.695704E -01                           2.610552E  00              1.150046E  01

   ITER            =              6
   DETER           =              0.49611E 07
                             XOLD(1)…XOLD(    7)
      4.569306E -01                          -4.071984E -04              -2.125199E -03
9.151720E-01
     -3.695703E -01                           2.610549E   00              1.150045E  01
SUCCESSFUL CONVERHENCE
   ITER            =              6
                             XOLD(1)…XOLD(    7)
     4.569306E -01                          -4.071984E -04               -2.125199E -03
9.151720E -01
    -3.695703R -01                           2.610549E  00               1.150045E   01
```

Tables (3.2) Equilibrium Gas mixture

x_1	Mole fraction CO	0.322871
x_2	Mole fraction CO_2	0.009224
x_3	Mole fraction H_2O	0.046017
x_4	Mole fraction H_2	0.618172
x_5	Mole fraction CH_4	0.003717
x_6	Mole fraction O_2 / CH_4	0.576715
x_7	Total moles of product	2.977863

In the feed gases, and total number of moles of product per mole of HC_4 in the feed are tabulated in Table (3.2). Thus the required feed ratio is 0.5767 moles of oxygen per moles of methane in the feed gases.

To establish if carbon is likely to be formed according to reaction (5.5.7) at 2200^0F for a gas of the computed composition, it is necessary to calculate the magnitude of

$$\vec{K} = \frac{P^2 co}{a_c P_{co2}} = \frac{Px_1^2}{a_c x_2} \qquad \ldots \qquad (3.25)$$

If $\vec{K}$ is larger than k_4 from (3.25), then there will be a tendency for reaction (3.24) to shift toward the left; carbon will be formed. Assuming that $a_c = 1$,

$$\vec{k} = \frac{20x(0.32287)^2}{1x0.009224} = 226.03 < k_4 = 1329.5. \quad - \quad - \quad - \qquad (3.26)$$

Therefore there will be no tendency for carbon to form.

† *Okereke, C. Emmanuel, Ph.D.*, Department of Mathematics, Michael Okpara University of Agriculture, Nigeria

REFERNCES

[1] Alefeld, G., Potra, F. A and Shen, z, on the existence theorems of kontorovich, more and mirands, preprint 2001.

[2] Ames, W., Nonlinear ordinary differential equations in transport processes. Academic press, New York 1968.

[3] Agyros, I. K. on an interative algorithon for solving non linear equations beritrage zir numerschen math, 10, 1 (1991) 83-92.

[4] Agyros, I. K., some generalized projection methods for solving operator equations, J. comp. Appl math 39, 1 (192) 1-6.

[5] A Ralston and H. S., wilf, Mathematical Methods for digital computers, Wiley, New York. (1960).

[6] Brice, carnahan, luther, H. A., Wilkes O. James. Applied numerical methods John Wiley and sons in cooperated, New York London. Sydney-Toronto.

[7] F. A. Jenkins and H. E. White fundamentals of optics 2nd Edition.

[8] J. F. traub, iterative methods for the solution of equations. Prentice-Hall, eagle wood cliffa. New Jessey 1964.

[9] Mc Graw – Hill, New York 1951.

[10] Ortega, J. m, and Rheinboldt, W. C. iterative solution of non linear equations in several variable academic press, New York, 1970.

[11] Rall, L. B. computational solution of non linear operator egns. Wiley New York 1968.