C.1

A COMPARISON OF SOLUTION METHODS FOR THE CHEMICAL EQUILIBRIUM PROBLEM

by

MARGARITA M. RUDA

Lic. en Quimica, Universidad de Buenos Aires, Argentina, 1973

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE
in
THE FACULTY OF GRADUATE STUDIES
(Department of Chemical Engineering)

We accept this thesis as conforming to the
required standard

THE UNIVERSITY OF BRITISH COLUMBIA
22 April 1982

© Margarita M. Ruda, 1982

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of CHEMICAL ENGINEERING

The University of British Columbia
1956 Main Mall
Vancouver, Canada
V6T 1Y3

Date AUGUST 31 1982

ABSTRACT

This thesis deals with computing the equilibrium composition of a multiple species reacting mixture. When pressure and temperature are constant and the system is ideal, this is the chemical equilibrium problem.

It is possible to approach this problem as the minimization of a non-linear objective function subject to linear equality constraints. The objective function represents the Gibbs' free energy of the system; the constraints refer to the conservation of the elements. Such a formulation corresponds to a "dual geometric program" which is related to another optimization problem known as the "primal geometric program". In those chemical equilibrium problems with many species, the "primal geometric programming" formulation includes less variables and constraints (inequality ones) than the dual formulation.

We first compared the primal and dual formulations of the chemical equilibrium problem. Both formulations were solved with a Generalized Reduced Gradient code on seven examples. The primal formulation proved to be 30% faster than the dual for middle-sized problems (up to six simultaneous reactions). The code failed when trying to solve a dual problem of 24 species and 4 elements.; but this same problem was easily solved when formulated as a primal geometric program.

As the geometric programming theory includes sensitivity analysis, and we were also interested in the effects of small changes of pressure and temperature on the optimal solution, we compared sensitivity analysis with re-optimization of the problem. Sensitivity analysis proved to be between 30% to 50% faster than re-optimization. It also yielded accurate results for the more abundant species when relatively small changes of temperature and pressure were operated. However, the equilibrium concentrations of trace species hardly matched those calculated by re-optimization.

From these results we recommend the use of the primal formulation and of re-optimization to solve the chemical equilibrium problem, and we present a computer code that has been tested on a variety of examples.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr D. W. Thompson for his patient help and understanding.

I am very grateful to Mike Patterson and the rest of the people of the Numerical Analysis group in the Computer Centre at UBC. They were very helpful with programming hints.

I also want to thank my husband, Luis Sancholuz, for his encouragement and his typing.

Grants from UBC-NAHS are gratefully aknowledged.

CHAPTER I: INTRODUCTION

The problem of finding the chemical composition of a reacting mixture when it reaches equilibrium has been called the "chemical equilibrium problem" (Dantzig _et_ _al_., 1958). In this problem, two thermodynamic variables, pressure and temperature, are fixed. Throughout this thesis,we will assume that the system is ideal.

Knowing the composition at equilibrium is important in applied fields as diverse as chemical reactors design, development of rocket propellants, evaluation of explosives or biochemistry. When few reactions take place,the calculations needed are quite trivial. They become very cumbersome, however, when there is a large number of chemical species and simultaneous reactions . If these calculations have to be repeated for slight changes in the fixed conditions (i.e., pressure and temperature), the problem can be quite impossible to solve in a reasonable period of time. A digital computer is then required.

This thesis aims at producing a general computer program to solve the chemical equilibrium problem for ideal systems of many chemical species. To avoid re-solving the problem when small changes in pressure and temperature need to be considered, the program should also be capable of performing sensitivity analysis. Eventually, this program will be used for simulation and design of chemical reactors, mainly for gaseous reactions.

To achieve our goal, we needed to address the following points: i) the mathematical formulation of the problem, ii)the numerical method of solution, and iii) the simplicity of the program for the user. In that order we now proceed.

The mathematical formulation of the problem has to be general enough to apply to a wide range of examples. The chemical equilibrium problem has been posed in many ways, all of them mathematically equivalent albeit numerically different. Van Zeggeren and Storey (1970) classified these formulations as the "free energy minimization" approach and the "equilibrium constant" method. The former is an optimization problem with a non-linear objective function and linear equality constraints. The latter can be derived from the condition of minimum Gibbs free energy of the system, but consists of a set of simultaneous non-linear equations to be solved.

Passy and Wilde (1968) found a relationship between the chemical equilibrium problem and Geometric Programming, a mathematical programming technique (GP hereinafter). The "free energy minimization approach" can be regarded as a "dual GP" problem to which corresponds a related "primal GP" formulation. The "primal GP" formulation has a smaller dimension than the "dual GP", and has been succesfully used to solve examples of the chemical equilibrium problem (Passy and Wilde, 1970; Dembo, 1976; Lidor, 1975).

The numerical method of solution is of course related to the mathematical formulation of the problem; however, the same formulation can be solved using different numerical methods. The selection of the numerical method of solution is important for the computational effectiveness of our code. In other words, we want to produce accurate results fast. Both the primal GP and dual GP formulations present numerical difficulties. There is not enough computer evidence in the literature as to which method of solving the chemical equilibrium problem performs better. In this work we shall compare the perfomances of the primal and the dual GP formulations when solving the chemical equilibrium problem. A general purpose non-linear optimization code, which can handle both formulations, seems appropiate to make that comparison possible.

GP theory provides ways of performing sensitivity analysis. This is particularly useful to us since we are interested in comparing its speed and accuracy with that of actually re-solving the problem. The results of the comparisons (primal vs. dual and sensitivity analysis vs. repeating the method) will permit us to choose the more appropiate mathematical formulation and numerical method of solution for a final computer code to solve the chemical equilibrium problem.

We finally come to the question of simplicity. We want to avoid the user's tendency to err while inputing data. The amount of information needed to run the program should be reduced to a minimum. A subroutine to calculate a first starting point for the optimization is therefore necessary.

The remainder of the thesis is organized as follows. Chapter II presents the different mathematical formulations of the chemical equilibium problem . Chapter III reviews and discusses the literature on the subject. Chapter IV deals with the difficulties found in trying to set up the programs, while Chapter V describes the programs actually written for the proposed comparisons. Chapter VI presents and discusses the results obtained with our programs in a series of examples taken from the literature. Finally, conclusions and recommendations are included in Chapter VII.

CHAPTER II: MATHEMATICAL FORMULATIONS OF THE PROBLEM

This chapter will serve as a means of defining the problem, as well as introducing the nomenclature. It is based on a literature review, and it is organized as follows. Section 1 deals with the thermodynamic relations that describe the chemical equilibrium problem. In section 2 we introduce the two traditional mathematical formulations known as the "Gibbs free energy minimization method" and the "equilibrium constant method" (Van Zeggeren and Storey,1970). In section 3 we present the geometric programming theory, and we discuss the relationship between this mathematical programming technique and the chemical equilibrium problem. In section 4 we discuss the basis for sensitivity analysis in geometric programming.

Thermodynamic relations

The thermodynamic basis of the "chemical equilibrium problem" has been extensively discussed in many textbooks (Denbigh, 1966, Kirkwood and Oppenheim, 1961). We will review the thermodynamic relations that will allow us to formulate mathematically our problem.

When temperature (T) and pressure (P) are chosen as independent variables for a thermodynamic system, the appropriate fundamental relation that completely describes the system is the one expressing the Gibbs free energy (G) in terms of P, T, and the composition variables. We will assume a system of K phases and N chemical species; each chemical species is potentially present in each phase. Then,

$$G = G(T,P, \delta_{jk}) \tag{2.1}$$

where

δ_{jk} = number of moles of species j in phase k

Since Gibbs free energy is an extensive property, we can rewrite eq. 2.1 as :

$$G = \sum_{j=1}^{N} \mu_{jk} \quad \delta_{jk} \tag{2.2}$$

where the chemical potential is an intensive property defined as follows:

$$\mu_{jk} = \left(\frac{\partial G}{\partial \delta_{jk}} \right)_{T,P,\delta_{ik}} \tag{2.3}$$

If we neglect all interactions among the phases, each phase contributes additively to the Gibbs free energy of the system. Then,

$$G = \sum_{k=1}^{K} Gk\ (T,P,\delta_{jk}) \qquad 2.4$$

Where

Gk = Gibbs free energy of phase k

Let us now take a look at the functional form of the chemical potential. The chemical potential of the species j in the phase k can be written in the following form: (Denbigh, 1966)

$$\mu_{jk}(T,P,\delta_{jk}) = \mu_{jk}^{o}\ (T,P)+RT\ \ln a_{jk} \qquad 2.5$$

where

μ_{jk}^{o} (T,P)= reference value for the chemical potential of the species j in the phase k.

R = universal gas constant

a_{jk} = activity of species j.

We will now assume that each phase is an ideal mixture. Moreover, we will consider each species as belonging to one phase; if one species belongs to more than one phase, it will be given a different number. Hence, k is no longer needed as a subscript for the number of moles. In real life, our problem is restricted to :

a) gaseous reactions,

b) reactions in pure condensed phases,

c) some biological models.

With these assumptions, the activity of one species in the gaseous phase is equal to the molar fraction of this species in the gas phase, times the total pressure. For the ideal condensed phases, the activity equals the molar fraction. In equations, if we define

$$\lambda_K = \sum_{j \epsilon k} \delta_j \qquad 2.6$$

Then the molar fraction of species j, Xj is:

$$Xj = \delta_j / \lambda_k \qquad 2.7$$

and, for an ideal gas phase

$$a_j = Xj \ P/1 \text{ atm.} \qquad 2.8$$

for pure condensed phases

$$a_j = Xj \qquad 2.9$$

we can rewrite equation 2.5 for the gas and for the pure condensed phases

$$\mu_j\ (T,P) = \mu_j{}^o\ (T) + RT \ln P + RT \ln (\delta_j\ /\ \lambda_K) \qquad 2.10$$

$$\mu_j\ (T,P) = \mu_j{}^o\ (T,P) + RT \ln (\delta_j\ /\ (\lambda_K) \qquad 2.11$$

where the superscript ^{o} refers to standard state. For the gas, it is the chemical potential of the species j as an ideal gas, at zero pressure. For the condensed phases, it is usually the chemical potential of the pure species j in the condensed phase at the same T and P.

We will now define the free energy coefficients Cj as:

$$Cj = (\ \mu_j^o\ (T)\ /RT) + \ln P/1 \text{ atm.} \qquad 2.12$$

for the gas phase, and

$$Cj = \mu_j^o\ (T,P)\ /\ RT \qquad 2.13$$

for the condensed phases. Replacing equations 2.10, 2.11, into 2.2, dividing by R T , and using 2.12 and 2.13, we get a convenient formulation of the Gibbs free energy for an ideal system of N species and K ideal phases, at T and P constant.

$$G/RT = \sum_{j=1}^{N} \delta_j\ (\ln \delta_j + Cj) - \sum_{k=1}^{K} \lambda_K \ln \lambda_K \qquad 2.14$$

We have now a working equation that describes Gibbs free energy in terms of the number of moles of the species present at equilibrium, the total number of moles per phase, the temperature, pressure, and the "free energy coefficients" Cj . Let us consider now the equilibrium conditions. The condition for equilibrium in a closed system described by Gibbs free energy (G) is that G is a minimum. For the ideal case of equation 2.14, this minimum exists and is unique. (Denbigh, 1966). The fact that G is a minimum at equilibrium implies that the variations of G produced by independent variations must be zero. But not all the variations in the number of moles are independent. They must satisfy the requirement that the total mass of each element is distributed among the different chemical species (Zeleznik and Gordon, 1968). We need a mathematical description of these constraints.

Formulations of the chemical equilibrium problem

The free energy minimization method

The "free energy minimization " method is just the mathematical formulation of the equilibrium conditions stated above. We still have to formulate the constraints, since the Gibbs free energy is described by equation 2.14. To do that, we define the "exponent matrix" AA . It is an M x N matrix, where M is the number of elements in the system, and N is the number of chemical species in the system. Each column of the matrix corresponds to the chemical formula of one species: AAi,j gram atoms of the element i are present in one mole of species j.

If the system contains Bi gram atoms of element i, then the conservation of elements can be written as:

$$\sum_{k=1}^{K} \sum_{j \in k} AA_{ij}\,\delta_j = B_i \qquad i = 1,M \qquad 2.15$$

If ionization is considered, the conservation of charge can be also expressed as in equation 2.15 ; charge is considered to be the M+1 element, with zero amounts; the corresponding row in the exponent matrix is the charge of each species.

Besides the conservative constraints, we have N positivity constraints, since the number of moles of a chemical species is either positive or zero. They are:

$$\delta_j \geqslant 0 \qquad j = 1, N \qquad 2.16$$

Combining the equations 2.14, 2.15, and 2.16, we have the "Gibbs free energy minimization " formulation of the chemical equililbrium problem. We will call this formulation Problem A.

Problem A

$$\text{Min } G/RT = \sum_{j=1}^{N} \delta_j (\ln \delta_j + Cj) - \sum_{k=1}^{K} \lambda_k \ln \lambda_k$$

$$\text{s.to:} \quad \sum_{k=1}^{K} \sum_{j \in k} AAij\, \delta_j = Bi \qquad i = 1,M$$

$$\delta_j \geqslant 0 \qquad j = 1,N \qquad 2.17$$

In short , Problem A is an optimization problem of:

a) non linear objective function of N variables (the number of chemical species present at equilibrium).

b) M linear equality constraints (number of elements).

c) N positivity conditions.

In spite of Gibbs developing his theory in the past century, it was not until recently that Problem A could be solved efficiently by computer optimization techniques. Among the difficulties, the constraints are equality ones; the objective function is convex for ideal systems, but it is non-differentiable if one of the species is zero, and it is not defined in such a case; the dimension of the problem is the number of species present at equilibrium, which can be a large number. The main advantage of Problem A is its simplicity of formulation; the only data needed are the exponent matrix , the amount of elements (B vector) and the free energy coefficients C, determined by the working pressure and temperature.

Reducing the free energy minimization problem

Problem A was too large to be solved when computers were not available; stoichiometry was used to simplify numerically the problem. The idea was to write some species (constituents) as a function of others (components) (Brinkley, 1946). A system of many constituents was regarded as being formed by the components through a set of simultaneous linearly independent chemical reactions. Just how many of these reactions should be considered, or which should be the components, has been the object of much research for the case of many constituents (Denbigh, 1966; Schubert and Hoffmann, 1976; Waller *et al*. 1980).

A well posed chemical equilibrium problem should have all its proposed species greater than zero at equilibrium; also the balances (eq. 2.15) should form a set of linear equations with rank = M. If this is the case, there exists a matrix U called the stoichiometric matrix, with dimensions N x D , and rank D, where D= N-M , is the number of independent chemical reactions. The stoichiometric matrix is such that, in matrix notation,

$$AA.U = 0 \qquad 2.18$$

Then the composition of each species can be written as a linear combination of D independent parameters r called the extent of the reactions. That is,

$$\delta_j(r) = \delta_j^{\,o} + \sum_{d=1}^{D} Ujd\; r_d \qquad j = 1,N \qquad 2.19$$

where

$\delta_j^{\,o}$ = initial amount of species j

Also see that:

$$\left(\frac{\partial \delta_j}{\partial r_d}\right)_{T,P} = Ujd \qquad 2.20$$

which is the traditional way of defining the extent of reaction. To calculate the stoichiometric matrix is easy when few reactions take place but it is not so for the case of many reactions. Some systematic ways of calculating it from the mass balances have been developed lately (Schneider & Reklaitis, 1975). In fact, any solution of the homogeneous system of equations formed by setting the B vector equal to zero in equation 2.15, is a stoichiometric matrix , even if it does not look so nice to a chemist. (Aris, 1970). The selection of the components is the more difficult task in all the systematic procedures. The best way to do it is to choose as components those species present in greater quantities at equilibrium, which is most of the time difficult to know beforehand. Isomers also present difficulties (Cavallotti et al. , 1980).

If the number of moles of the species are written as in equation 2.18, then the free energy of the system can be stated as a reduced problem of D variables, (the extents of each reaction) with D positivity constraints. To simplify the expression resulting by replacing 2.18 in 2.14 and 2.15, we introduce the following constant terms:

$$-\ln Ko = \frac{\sum_{j=1}^{N} \delta_j^o \, Cj}{RT} \qquad 2.21$$

$$-\ln Kd = \frac{\sum_{j=1}^{N} Ujd \, Cj}{RT} \qquad 2.22$$

We now define:

$$\lambda_K(r) = \lambda_K{}^o + \sum_{d=1}^{D} r_d \, \lambda_{Kd} \qquad k = 1,K \qquad 2.23$$

$$\lambda_K{}^o = \sum_{j \epsilon k} \delta_j{}^o \qquad k = 1,K \qquad 2.24$$

$$\lambda_{Kd} = \sum_{j \epsilon k} Ujd \qquad d = 1,D \quad k = 1,K \qquad 2.25$$

After some algebra we get the reduced free energy minimization formulation, which we will call Problem B.

Problem B

$$\text{Min } G/RT = -\ln Ko + \sum_{d=1}^{D} r_d \ln Kd + \sum_{j=1}^{N} \delta_j(r) \ln \delta_j(r) - \sum_{k=1}^{K} \lambda_K(r) \ln \lambda_K(r)$$

s. to

$$\delta_j(r) = \delta_j^0 + \sum_{d=1}^{D} r_d \, Ujd \geqslant 0 \qquad j = 1,N \qquad 2.26$$

Problem B is an optimization problem, with

a) non-linear objective function of D variables, which are the extents of each independent reaction.

b) N positivity conditions, one per species present at equilibrium.

In this formulation, the number of variables is smaller than in the previous one; but when N>2M , this advantage is not so important. The objective function is still undefined and non differentiable when a variable is equal to zero. The set of stoichiometric coefficients has to be calculated before the optimization procedure.

Formulation of the equilibrium constant method

Problems A and B are two forms of the optimization approach to the chemical equilibrium problem. In order to derive the "equilibrium constant" approach, we will consider Problem B and the conditions for optimality. If G is a minimum at equilibrium, with pressure and temperature constant, then the gradient of G has to be zero at this point. Taking the gradient of G, expressed as a function of the extents of reactions, and setting it equal to zero, we get a set of D non-linear equations in D unknowns. These equations are the well known "mass action law".

"Problem C" below states the equations.

<u>Problem C for gaseous phase</u>:

$$Kd = \exp - \left| \frac{\sum_{j \epsilon k} Ujd\ Cj}{RT} \right| =$$

$$= \prod_{j \epsilon k} \left| \frac{(\delta_j^{\,o} + \sum_{d=1}^{D} Ujd\ r_d)}{\sum_{j \epsilon k} (\delta_j^{\,o} + \sum_{d=1}^{D} Ujd\ r_d)} P \right|^{Ujd}$$

$k = 1 \quad d=1,D$ 2.27

Problem C for condensed phases

$$K = \exp -\left| \frac{\sum_{j\epsilon k} Ujd\ Cj}{RT} \right| =$$

$$= \prod_{j\epsilon k} \left| \frac{(\delta_j^{\,o} + \sum_{d=1}^{D} Ujd\ r_d)}{\sum_{j\epsilon k} (\delta_j^{\,o} + \sum_{d=1}^{D} Ujd\ r_d)} \right|^{Ujd}$$

$k = 2,K \quad d = 1,D$ 2.28

Stoichiometric coefficients are needed in Problem C. Once the set of reaction variables r that satisfies the equations is known, the number of moles present at equilibrium is calculated through equations 2.19. Problem C involves solving a system of non linear equations. This is a totally different numerical approach than problems A and B, although the mathematical formulations have been shown to be equivalent.

Almost all formulations of the chemical equilibrium problem in the literature fall into Problems A, B, or C, with some algebraic modifications. There is still a different formulation, and to introduce it we need some background on the mathematical programming technique known as Geometric Programming.

Geometric programming theory

Geometric Programming is the mathematical formulation of a special kind of optimization problem. It involves the minimization of a posynomial (positive polynomial) objective function g , subject to inequality constraints that are also posynomials. The problem described is called a "primal problem" ; the primal function g belongs to the Euclidean space R+m there exists a related "dual" maximization problem, involving a function v that belongs to the dual space R+n. The function v has the form of a product of non-linear terms, and has M+1 linear constraints. It has been found that the constrained maximum value of v is equal to the constrained minimum value of g (Duffin, Petersen and Zener, 1966). We will now present the equations that exemplify all this. A Primal Geometric Program is of the form:

Primal GP

Min $g_o(t)$

s. to

$$g_k(t) \leqslant 1 \qquad k=1,K$$
$$t_i \geqslant 0 \qquad i=1,M$$

where

$$g_k(t) = \sum_{j \in k} c_j \prod_{i=1}^{M} t_i^{A_{ij}} \qquad k=0,K \quad j=1,N \qquad 2.29$$

The corresponding maximizing Dual Geometric Program (Dual GP) is as follows:

Dual GP

$$\text{Max } v(\delta) = \prod_{j=0}^{N} (c_j / \delta_j)^{\delta_j} \prod_{k=1}^{K} \lambda_k^{\lambda_k}$$

s. to

$$\sum_{j\in 0} \delta_j = 1 \qquad \text{normality condition}$$

$$\sum_{j=0}^{N} A_{ij}\,\delta_j = 0 \qquad i = 1,M \quad \text{orthogonality conditions}$$

$$\delta_j \geq 0 \qquad j = 0,N \quad \text{positivity conditions}$$

$$\text{where} \quad \lambda_k = \sum_{j\in k} \delta_j \qquad k = 1,K \tag{2.30}$$

The relation between the primal and dual problems is called the "geometric inequality" and states:

$$g\,(t) \geq \min g(t) = \max v(\delta) \geq v(\delta) \tag{2.31}$$

When the equality holds at the optimum (*), the primal and the dual variables are related through the following set of equations (Duffin, Petersen & Zener, 1966) :

$$\ln c_j + \sum_{i=1}^{M} Aij \ln t_i^* = \ln (\delta_j^* \, g_o(t^*)) \qquad j \epsilon k = 0 \qquad 2.32$$

$$\ln c_j + \sum_{i=1}^{M} Aij \ln t_i^* = \ln (\delta_j^* / \lambda_k^*) \qquad j \epsilon k = 1,N \qquad 2.33$$

If in the primal problem the constraints are active, and if it is solved by means of a Lagrangian technique, there is a relationship between the Lagrange multiplier for the constraint and the corresponding total number of moles of the phase :

$$\lambda_k = \Pi_k / g_o \qquad 2.34$$

where Π_k = Lagrange multiplier of the k-primal constraint

We will now summarize some important aspects:

a) Each primal variable is associated to one of the orthogonal conditions in the dual.

b) Each primal constraint is associated to one of the λ values.

c) Each primal term is associated to one dual variable.

d) The posynomial terms in the primal objective function correspond to the dual variables subject to the normality condition.

We will not go any further into Geometric Programming theory for the moment; we will discuss instead the relationship between

Geometric Programming and the chemical equilibrium problem.

Analogies between GP and the chemical equilibrium problem

In 1968 Passy and Wilde found that the chemical equilibrium problem, stated as our Problem A , could be regarded as a dual geometric program. Problem A was algebraically transformed through the following definitions to fit into the standard formulation of a dual geometric program.

$$\delta_o = 1 \qquad 2.35$$

$$c_j = \exp(-Cj) \qquad 2.36$$

$$c_o = 1 \qquad 2.37$$

$$-Bi = Aio \qquad 2.38$$

Since exp(-G/RT) is a monotonic function, finding its maximum is equivalent to minimizing the negative of the function; taking logarithms, we have:

$$\text{Min } (G/RT) \equiv \text{Max } \exp(-G/RT) \qquad 2.39$$

The Dual GP chemical equilibrium problem is then:

Chemical equilibrium as a dual GP

$$\text{Max } v = \exp(-G/RT) = \prod_{j=0}^{N} (c_j/\delta_j)^{\delta_j} \prod_{k=1}^{K} \lambda_k^{\lambda_k}$$

s. to

$\delta_o = 1$ normality condition

$\sum_{j=0}^{N} Aij\ \delta_j = 0 \quad i = 1,M$ orthogonality conditions

$\delta_j \geqslant 0 \quad j = 0,N$ positivity conditions

where

$\lambda_K = \sum_{j\epsilon k} \delta_j \quad k = 1,K$ 2.40

If we maximize the logarithm of the previous problem, we obtain a "transformed dual" formulation which is similar to our "Problem A". The matrix A is the augmented exponent matrix (AA); δ_o is a dummy species.

Transformed dual GP

$$\text{Max } \ln v \equiv \text{Min } G/RT = \sum_{j=0}^{N} \delta_j (\ln \delta_j + C_j) - \sum_{k=1}^{K} \lambda_K \ln \lambda_K$$

s. to

$$\delta_o = 1 \qquad \text{normality condition}$$

$$\sum_{j=0}^{N} A_{ij}\, \delta_j = 0 \qquad i = 1,M \qquad \text{orthogonality conditions}$$

$$\delta_j \geqslant 0 \qquad j = 0,N \qquad \text{positivity conditions}$$

where

$$\lambda_K = \sum_{j \in k} \delta_j \qquad k = 1,K$$

2.41

From the theory of Geometric Programming, to the previous dual problem corresponds the following primal problem:

Chemical equilibrium as primal GP

$$\text{Min } g_o(t) = \prod_{i=1}^{M} t_i^{A_{io}}$$

s. to

$$g_k(t) = \sum_{j \in k} c_j \prod_{i=1}^{M} t_i^{A_{ij}} \leqslant 1 \qquad k = 1,K$$

$$t_i \geqslant 0 \qquad i = 1,M$$

where Aio=-Bi

2.42

To the "transformed dual" corresponds a "transformed primal" formulation; we define $Z = \ln t$ and then:

<u>Transformed primal</u>

$$\text{Min } \ln g_o(t) = \text{Min } h(Z) = \sum_{i=1}^{M} Aio\ Zi$$

s. to

$$\sum_{j \epsilon k} \exp(Cj + \sum_{i=1}^{M} Aij\ Zi) \leqslant 1 \qquad k = 1,K \qquad 2.43$$

The primal GP chemical equilibrium problem is a new formulation of the problem, with:

a) minimization of a non-linear objective function of M variables (the number of elements)

b) K non-linear inequality constraints, one per phase. The constraints are active at the optimum. Each term of the constraints is equal to the molar fraction of the corresponding species in the phase; hence the constraints state that the summation of the molar fractions per phase has to be one at equilibrium. The total number of moles per phase can be calculated from eqn. 2.34

The transformed primal involves:

a) minimization of a linear objective function of M variables

b) K non-linear inequality constraints, with the same physical meaning as in the primal problem. If the transformed primal is solved using the Lagrange multipliers technique, each Lagrange multiplier is equal to the total number of moles in that phase.

The reduced dual

The same methods described before to derive Problem B from Problem A are used in the Geometric Programming theory. Problem B is thus equivalent to a "Reduced Dual Geometric Program" , except for some changes in nomenclature. These changes are shown in Table II-1.

Summary of the mathematical formulations

Table II-2 summarizes the most important features of the mathematical formulations of the chemical equilibrium problem.

We will now take a look at one convenient derivation from the Geometric Programming theory : the post-optimal analysis known as sensitivity analysis.

Sensitivity analysis

All the formulations of the chemical equilibrium problem are depending on the free energy coefficients C, the exponent matrix AA and the vector of the quantity of the elements, B. The exponent matrix will not change if the model is well formulated. The free energy coefficients will probably vary with the source from where they are obtained, but more important for practical purposes is their variation with the temperature and pressure of the system. The B vector may vary with different initial compositions, for example.

Table II-1. A parallel between Geometric Programming and Chemical Equilibrium nomenclatures.

Symbol	Chemical Nomenclature	GP Nomenclature
r	Extent of reaction	Basic variable
δ	Number of moles of species j	Dual variable
k=0	Not defined	Corresponds to primal objective function
δ_0	Not defined	Normality condition
Ujd	Stoichiometric coefficients	Nullity vectors
Kd	Equilibrium constant for the d reaction	Basic constant

Sensitivity analysis is just a way of evaluating the changes in the equilibrium Gibbs free energy and in the composition, when any of the parameters mentioned above is changed, without actually re-solving the optimization problem. The method involves a numerical calculation of the partial derivatives at the optimum, that is, the truncation of a Taylor series expanded around an already found optimum. Of course, if the variations in the parameters are large enough, the problem has to be re-solved.

The problem of obtaining a set of sensitivity equations has been approached in two ways in the literature. Both approaches

Table II-2. A summary of the more important characteristics of the formulations of the chemical equilibrium problem.

Name of the formulation in this work	Eqn.	Type of numerical solution	Number of variables	Number and type of constraints	Need stoich. coeff.
Problem A	2.17	Optimization	N	M linear eq. const. N positivity conditions	No
Problem B	2.26	Optimization	D	D nonlinear ineq. const.	Yes
Problem C	2.27 2.28	System of D nonlinear equations	D unknowns	-	Yes
GP dual and transformed dual	2.40 2.41	Optimization	N+1	M+1 linear eq. const. N positivity conditions	No
GP reduced dual	2.26	Optimization	D	D nonlinear constraints	Yes
GP primal	2.42	Optimization	M	K nonlinear ineq. const. M positivity conditions	No
GP transformed primal	2.43	Optimization	M	K non linear ineq. const.	No

Eqn.:equation, stoich. coeff.: stoichiometric coefficients, . eq.:equality, ineq.: inequality, const.: constraints, N: number of species, M: of elements, K : of phases, D: of reactions.

differ from the numerical point of view.

The first approach was stated in appendix B of Duffin, Petersen and Zener's book (1966). They work with the Reduced Dual GP (eqn. 2.26). The Jacobian matrix of the transformation from the r variables to the Kd "variables"(matrix J) is formed as follows :

$$J_{qs}(\delta) = \sum_{j=1}^{N} (U_{jq} U_{js} / \delta_j^*) - \sum_{k=1}^{K} (\lambda_{kq} \lambda_{ks} / \lambda_k^*) \qquad 2.44$$

$$q,s = 1,D$$

where

$$\lambda_{kq} = \sum_{j \in k} U_{jq} \qquad 2.45$$

$$\lambda_{ks} = \sum_{j \in k} U_{js} \qquad 2.46$$

$$\lambda_k^* = \sum_{j \in k} \delta_j^* \qquad 2.47$$

Duffin et al. (1966) proved that the matrix J is also the Hessian matrix for the function ln v relative to the basic variables r (extents of reactions). So, if the reduced dual GP equivalent to the Problem B is solved by any method involving derivatives, the matrix J is readily available. The matrix J evaluated at the optimum point is used both to introduce criteria for the existence of derivatives and to derive expressions for these derivatives. The differential changes are approximated linearly.

Dinkel and Lakshmanan (1976, 1977) used these expressions in the case of the chemical equilibrium problem when P, T, and the amount of elements changed. They applied their results to two examples, the problems 3 and 5 of appendix B. They also suggested the use of an incremental procedure in order to control the error produced by the linear approximation of the differential changes. The expressions they obtained are as follows:

$$\frac{\Delta v}{v^*} \quad \sum_{j=1}^{N} \delta_j{}^* \frac{\Delta c_j}{c_j} \qquad 2.48$$

$$\Delta\delta_j = \sum_{d=1}^{D} \left\{ Ujd \sum_{q=1}^{D} \left(Jdq^{-1}(\delta_j^*) \sum_{j=1}^{N} Ujq \frac{\Delta c_j}{c_j} \right) \right\} \qquad 2.49$$

The expressions are valid if J is non singular at the optimum. As seen from equations 2.48 and 2.49, the inversion of the matrix J evaluated at the optimum is a necessary step to perform sensitivity analysis with this approach. When the number of reactions D is not very large, this is not a problem; but as the dimension of the matrix increases, so do the rounding errors and the time that is consumed in the inversion.

The second approach taken was related to Generalized Geometric Programming. This is an extension of the Geometric Programming theory to "generalized polynomials", that is polynomials with some negative terms. These researchers also tried to obtain expressions for numerical derivatives, but their approach was slightly different than the previous one (Rijckaert, 1974) . The M+1 dual constraints were written in explicit form, over the N+1 variables , as well as the D "equilibrium conditions" and the K equations relating λ to the summation of the number of moles in each phase. All this is equivalent to writing the Kuhn Tucker conditions for the dual GP chemical equilibrium problem. They wrote the N+1+K set of equations for the optimum and for a small perturbation of the optimum. Each equation in the perturbed set was subtracted from the corresponding one in the optimal set, and a Taylor's approximation series was used to linearize the equations. The final system of N+K+1 linear equations in N+K+1 unknowns(the variations in the number of moles) is as follows:

$$\Delta \delta_0 = 0 \qquad 2.50$$

$$\sum_{k=0}^{K} \sum_{j=0}^{N} Aij\ \Delta\delta_j = 0 \qquad i = 1,M \qquad 2.51$$

$$\sum_{j \epsilon k} \Delta\delta_O - \Delta\lambda_K = 0 \qquad k = 1,K \qquad 2.52$$

$$\sum_{k=0}^{K} \sum_{j \epsilon k} (Ujd/\delta_j{}^*)\ \Delta\delta - \sum_{k=0}^{K} (Ujd/\lambda_k{}^*)\ \Delta\lambda_K$$

$$= \sum_{k=0}^{K} \sum_{j=1}^{N} Ujd(\ln c_j - \ln c_j{}^*)$$

$$d = 1,D \qquad 2.53$$

For any changes in the free energy coefficients, only the right hand side of eq. 2.53 changes. For any changes in the B vector, only the first column of eq. 2.51 changes. No inversion of a matrix is required, but the dimensions of the system are much larger than in the first approach.

CHAPTER III: LITERATURE REVIEW

This review of the literature is organized in three sections. First we take a look at the literature on general methods of solution for the chemical equilibrium problem. In the second section, we focus on the computer codes available for solving geometric programs. The third section deals with the geometric programming approach to the chemical equilibrium problem.

General aspects of the problem

In the previous chapter we have presented different formulations of the chemical equilibrium problem. They are all mathematically equivalent, but different numerical methods of solution are used for each case. The methods of solution fall into two main categories:

a) optimization methods

b) solution of non-linear systems of equations.

The problem now seems to be reduced to a choice of a mathematical formulation and a numerical method of solution. The literature on the possible combinations is very wide; an extensive review was conducted by Van Zeggeren and Storey (1970) and by Zeleznik and Gordon (1968). However, the problem is not yet settled (Cavallotti *et al.*, 1980).

Two factors should be pondered when judging the goodness of a solution method: the purpose of the calculations, and the means that are available to carry them on. We are not as much interested in the best solution for one specific case, as we are in a method flexible enough to accommodate a wide range of chemical equilibrium examples. The calculations will be performed by a digital computer. At the moment, there are two general computer codes available commercially that solve the chemical equilibrium problem. They belong to the RAND Corporation (1965,1970) and to the NASA (1971) respectively. We will briefly describe their methods.

The RAND method

RAND researchers devised the first version of the RAND program in 1958 (Dantzig, Johnson, White and De Howen, 1958). The program was revised and completed by R.J.Clasen, N. Shapiro, M. Shapley and others over a period of 15 years.

The chemical equilibrium problem is formulated as a minimization of the Gibbs free energy subject to mass balance constraints. The formulation is similar to Problem A of Chapter II in this thesis; it deals with an ideal gas phase and pure condensed phases.

The solution is obtained in two steps. The "first order method" provides a first set of composition values through a simplification of the problem to a linear program. The "second order method" uses the previous set of values as a first guess. The Gibbs free energy is approximated at this point using a second order Taylor's series. Then, the problem is transformed to an unconstrained optimization, using Lagrange multipliers. A final set of equations is then solved using the Newton-Raphson technique.

The RAND program has proved successful over a number of years in solving the chemical equilibrium problem. The main disadvantages, however, are:

a) slow convergence when many trace species are included in the model

b) inability to handle cases where the number of moles of a species is zero.

The NASA method

The first NASA programs to calculate chemical equilibrium composition were based on the "chemical equilibrium constant" formulation. The method was first devised by Huff (1951) and was later modified by Zeleznik and Gordon (1960,1962). However, in 1971, they changed the method to a free energy minimization procedure;the reasons given for the change included (NASA , Manual Report,1971):

a) more bookkeeping necessary for the equilibrium constant method.

b) numerical difficulties with the use of components (compared with keeping all the variables as such).

c) more difficulty in extending the generalized method for non-ideal equations (but still the program handles only ideal gases and pure condensed phases).

It was shown by Gautam and Seider (1979) that the RAND and NASA methods,although derived differently, give nearly identical equations and are both implementations of Newton's method. The

NASA code calculates thermodynamic derivatives as well as the chemical composition.

We have already stated that the chemical equilibrium problem can be regarded as a geometric program. Let us now look at some algorithms devised to solve geometric programming problems.

Algorithms to solve geometric programs

Many algorithms to solve geometric programs have been proposed in the literature. A summary of some of these can be found in Bleightler and Phillips' book(1976). We may classify the algorithms in three groups.

a) algorithms that solve the primal geometric program (GP),

b) algorithms that solve the dual GP,

c) general non-linear optimization methods that can be applied to solve either the primal or the dual GP.

For the general case, it is not possible to predict if the dual or the primal problem are easier to solve. Empirical evidence based on computational comparison is needed. Some comparisons among codes were performed by Dembo (1978),Sarma, Martens et al (1978) and by Gochet, Loute and Solow (1978). We will give a brief description of some of the GP and general purpose optimization codes compared and we will then try to summarize the conclusions of the papers mentioned above.

Algorithms that solve the primal

The primal GP was stated in equation 2.29 (chapter II). The chemical equilibrium problem as a primal GP was described by equation 2.42.

If we minimize instead the logarithm of the objective function, and if we make the transformation of variables z = ln t, then the problem becomes a convex program. The t's have to be positive,but the z's are unrestricted in sign. The resulting problem is called the transformed primal problem,and was specified in equation 2.43 for the chemical equilibrium problem.

Another way of solving the primal is via separable programming techniques; this formulation increases the dimensionality of the problem from M to N.

Based on the previous formulations, the algorithms that solve the primal can be classified in three groups.

1) Condensation : the primal geometric programming is solved directly by condensation or linearization of posynomial functions (cutting planes algorithms). A well known code based on this approach is GGP. Written by Dembo (1974), it is based on a Kelley's cutting plane algorithm.

2) Kuhn-Tucker conditions for optimality. The Kuhn Tucker conditions for primal geometric programming are solved iteratively, using a condensation technique. One code based on this approach is GPKTC, written by Rijckaert and Martens (1976). This method is essentially equivalent to a Newton-Raphson algorithm for the Kuhn-Tucker conditions expressed in terms of the variables z=ln t. The code FP from Gochet, Loute and Solow,

is based on the same approach

3) Separable Programming. A good code based on this formulation is DAP, by G.V.Reklaitis ; he uses the differentiable algorithm of Wilde and Bleightler (Dembo, 1978).

Algorithms that solve the dual

The dual geometric programming (DGP) is the linearly constrained, non-linear programming problem stated on equation 2.30 (chapter II); equation 2.40 for the case of chemical equilibrium.

Taking logarithms of the objective function, the problem is transformed to a concave objective function to be maximized, subject to a linear equality constraints. For the chemical equilibrium, that transformation is seen in equation 2.41.

The reduced dual geometric program (RDGP) is obtained by eliminating M+1 basic variables from the program DGP and expressing them in terms of D= N+1-(M+1) nonbasic variables (see eqn. 2.26, 2.27 and 2.28 on chapter II) most of the algorithms that solve the dual make use of the reduction of dimensionality that the RDGP problem provides.

The main problems related to solving the dual are (Gochet, Louter, and Solow, 1974):

a) non differentiability of the objective function with respect to the dual variables where they take on the value zero.

b) the dimensionality of the dual problem will always be larger than that of the primal, except for cases with N<2M solved using the reduced dual GP.

c) dual variables have to be determined with higher accuracy if primal variables are to be calculated from them.

The algorithms that solve the dual problem are based on the following principles:

1) Linear approximation methods The dual program in its logarithmic form is approximated at a specific point by a linear program (LP). The solution of the LP provides a direction for improving the value of the objective function. The step is provided by a line minimization. One code that follows this approach is LAM, from Rijckaert and Martens (1978): the LP routine is based on the Revised Simplex Method with product form of the inverse. The line minimization is based on cubic interpolation.

2) Separable programming. After a logarithmic transformation, the dual objective function is separable and can be approximated by a piecewise linear function. One code that follows this approach is SP, by Rijckaert and Martens (1978)

3) Gradient projection methods. This approach combined the gradient projection method due to Rosen with a variable metric method in order to approximate the inverse of the Hessian of the objective function. One example is the code VMP (Sargent and Murtagh, 1973).

4) Newton-Raphson. The Kuhn-Tucker conditions for the reduced dual geometric program result in a system of non linear equations that are solved using a Newton-Raphson technique (see the similarities with the "equilibrium constant" approach in the chemical equilibrium problem). Many codes are based on this

approach, like NEWTGP by J. Bradley (1973) (that code has the capability of performing sensitivity analysis), GEOGRAD by Dinkel and Kochenberger (1974), and NRF by Rijckaert and Martens, (1978).

5) Other methods

The code GOMTRY by Blau and Wilde (1971) solves the Kuhn Tucker conditions for the separable dual program ; the code CSGP by Beck and Ecker (1978) applies the concave simplex method to the dual; the code MCS introduces a modification to CSGP that allows for blocks of variables to go to zero simultaneously.

General purpose non-linear optimization algorithms

Many important algorithms for solving non-linear programs have been developed and refined in the last 10 years (Lasdon,1981). The more successful are :

1) Penalty function methods. The essential idea is to transform the general non-linear problem into a sequence of unconstrained problems. The more robust code is: SUMT (Carroll, 1959, Fiacco and Mc Cormicke, 1966). The objective function and inequality constraints can be non-linear functions but if there are equality constraints they have to be linear. Hence the code may be used to solve either the primal or the dual GP (Himmelblau, 1972). Another code , CONMIN, transforms a non-linear program with inequality constraints into an unconstrained problem using a penalty function method, and then the unconstrained problem is solved by Fletcher and Powell's algorithm (Haarhof and Buys, 1970).

2) Generalized Reduced Gradient methods(GRG). The GRG algorithm uses the linear equality constraints to accomplish the equivalent of algebraically eliminating an equal number of dependent variables from the problem. This reduces the problem to one with exclusively decision variables. This reduction affects the evaluation of the gradient. The reduced gradient is the gradient of the objective function with respect to the independent variables, subject to moving the independent variables in such a way that the equality constraints are satisfied (Westerberg, 1981). The reduced gradient is used to determine the direction of search. Hence, when GRG solves the dual GP problem, it really works with the "reduced dual problem", although the user should present the dual GP in its standard form.

When dealing with inequality constraints, GRG converts them into equalities by introducing slack variables. If the constraints are non linear, they are replaced by their second-order Taylor series approximates expanded at the point of interest. So GRG may also be used to solve the primal problem. UBC has a 1975 version of the code available.

3) Succesive Linear Programming (SLP). The Successive Linear Programming codes linearize any non-linear objective or constraint functions around a point, and then use the resulting linear program using efficient LP codes as subroutines. Some codes have been proposed by Griffith and Stewart (1961) and by Busby (1974)

Computational experience on GP codes and general non-linear codes for GP

To decide which is the best algorithm to solve geometric programs would involve trying all of them on a wide variety of examples; the algorithms should be codified by the same programmer, and they should be run with the same compiler on the same computer. The evaluation can then be made on the basis of time. This approach, however , seems quite impractical. The number of proposed algorithms is quite large; so is the range of problems to be tested. To be able to compare codes in different computers and with different compilers, standard times were defined. They refer to the execution time of the problem divided by the time required to execute Colville's timing program (Himmelblau, 1972). They still vary with the programmer, and sometimes with the compiler, and they are not always used in the literature.

We will now show the code comparisons done by a series of authors on different examples of geometric programs. We will focus on their conclusions when solving geometric programs that are also chemical equilibrium problems.

Sarma, Martens, Reklaitis and Rijckaert (1978) tested 5 algorithms to solve 16 GP problems . The codes were: SUMT on primal, GGP on primal, CSGP on dual, MCS on dual,and DAP on transformed primal. The basis for each code was given in the previous section in this chapter. GGP got the best results as for CPU times. SUMT had the largest times. They concluded that there was no definite evidence that solving the dual GP was computationally more attractive than solving the primal as has been stated by Duffin and Petersen (1967).

Rijckaert and Martens (1978) also performed computational comparisons on GP algorithms. They tried 16 GP codes (all the ones we have mentioned before and three more) and one general purpose non-linear program, CONMIN , on 24 problems . Their problem 4 is an scaled version of a primal chemical equilibrium as stated in Dembo (1976)--see prob.4, appendix B. On this particular problem the results of CPU times of the best algorithms are reported in table III-1.

Table III-1. Best CPU times (in sec.) for the algorithms compared by Rijckaert and Martens (1978) on the chemical equilibrium problem.

Problem	GPKTC (primal)	GGP (primal)	NRF (red. dual)	CONMIN (primal)
Problem 4	1.0	5.82	7.73	12.73

For the general case, in this comparison, GGP was the best code. But GPKTC worked better for the chemical equilibrium problem.

Dembo(1978) compared six GP codes and five general non-linear optimization codes on 8 problems. Problems 1A and 1B of his series are, respectively, the unscaled and scaled versions of the primal chemical equilibrium problem (prob. 4, appendix

B),already compared by Rijckaert and Martens. The results for the scaled problem are shown in table III-2.

Table III-2. Best standard times (in sec.) for the algorithms compared by Dembo (1978) on the chemical equilibrium problem.

Problem	GPKTC (primal)	GGP (primal)	GEOGRAD (dual)	GRG (primal)
Problem 1B	.0554	.271	.0565	.0560

The time relationship between GPKTC and GGP is similar, but not exactly the same as in the previous comparison. There is not much difference between GPKTC , the dual-based GP code GEOGRAD , and the general- purpose non-linear code GRG . Only GPKTC performed in the badly scaled problem.

Gochet,Loute and Solow (1974) tried one GP algorithm for solving the primal (FP), one GP algorithm for solving the dual (CSGP modified) and GRG on the primal GP on 16 problems. Problems 3 and 8 of their set are examples of the chemical equilibrium problem. Problem 3 is our problem 1 in Appendix B. It has 3 elements, 10 species and one phase; problem 8 has 6 elements, 16 species and 3 phases. The results are shown in table III-3.

Table III-3 . Best CPU times (in sec.) for the algorithms compared by Gochet, Loute and Solow (1974) on two chemical equilibrium problems.

Problem	CSGP (dual)	FP (primal)	GRG (primal)
3	1.68	.98	2.90
8	4.55	4.36	3.34

On the smaller problem, the primal based GP code was faster. GRG performed better on the bigger problem. Another conclusion of their work was that the application of GRG to the primal t-form of geometric programs was more efficient than using the ln-transformed variables. This conclusion was valid when the number of primal variables was less than 5.

Ratner, Lasdon and Jain (1978) compared the perfomance of GRG on the Dembo set of problems (1976) with that of Dembo's code. They also compared GRG on the 24 problems given by Rijckaert and Martens with the best times given in their paper. Standard times were used as a comparison. They reproduced the results on Dembo's paper already mentioned. They also pointed out the sensitivity of GRG to some tolerances that affect its perfomance. They are the tolerances for the objective function (stopping criteria) and for binding constraints, and the use of quadratic or tangent approximations of the initial values of the basis.

Finally, Eckes,Gochet and Smeers (1978) discussed the difficulties of solving the dual GP with GRG. They modified GRG using Beck and Ecker modification for the concave simplex in CSGP to account for the non-differentiability of the objective function at a point where some of the variables are zero, and to allow for some variables to become zero at optimality. The modified GRG was compared with the CSGP code. Half the number of iterations were needed with GRG than with the CSGP in most of the cases.

From these results, we may conclude that:

a) It is not always clear when the dual or the primal GP should be used.

b) For general cases, the best GP codes seem to be GGP (primal) and CSGP (dual). However, for the chemical equilibrium problem, GPKTC (primal) and GEOGRAD (dual) give better results in terms of time.

c) The general purpose non linear code GRG compares well with the best GP codes when solving geometric programming problems, specifically chemical equilibrium's scaled primal. The code can be used to solve either the primal or the dual GP .

d) In a general case, GRG seems to work better when solving primal geometric programs on the t-variables. However, the GP primal codes that performed better on the chemical equilibrium problems worked with transformed primal variables.

Geometric programmming and the chemical equilibrium problem

As we pointed out in the previous chapter, the chemical equilibrium problem reduces to a special case of geometric programming (GP). If we translate into GP nomenclature the traditional ways of formulating the chemical equilibrium problem, we can see that they all attempt to solve a dual GP, either as a problem with all its variables explicit or as a reduced problem .

Passy and Wilde (1968) devised a primal algorithm to solve the chemical equilibrium problem with only one ideal phase. It involved the formation of an unconstrained problem using Lagrange's method. They tried it on a hydrazine combustion problem with one ideal phase, 10 species and 6 elements (see problem 1 , appendix C). The algorithm worked with the primal variables t, which were scaled between 0 and 1. The objective function was elevated to the one tenth.

The RAND corporation model of the chemistry of the human respiratory system was used as a test problem by Dembo in a set of problems intended to compare GP codes (Dembo, 1976). We saw the results in the previous section. Bleightler and Phillips give an extense discussion of this particular problem solved by GGP (1976).

Dinkel and Lakshmanan (1975,1977) were interested on sensitivity analysis applied to the chemical equilibrium problem. They solved the dual problem using Dinkel and Kochenberger's GP algorithm (GEOGRAD code).

Finally Lidor (1975) devised a modification of the GGP algorithm to solve the primal chemical equilibrium problem. He worked with the transformed primal variables z= ln t, and used a Zangwill's cutting plane algorithm. His code includes the generation of a first starting point. He tried the algorithm on seven chemical equilibrium test problems, but his CPU timings compared disfavourably to the RAND code. The RAND method, as we pointed out earlier in this chapter, is basically a dual-based algorithm, and the RAND code has been perfected over many years. The question remains as if Lidor's code was slow because of the algorithm itself, or because of its implementation.

CHAPTER IV: WRITING THE PROGRAM--PRELIMINARIES

In order to write a general computer program to solve the chemical equilibrium problem we had to decide which mathematical formulation and which numerical technique of solution should be used. As for a mathematical formulation, we decided to use the geometric programming approach, because it provided a new form of presenting the problem (the primal formulation), and it gave the basis for a sensitivity analysis. We wanted to compare the primal and dual formulations of the geometric program, since no definite conclusions about the superiority of either one could be drawn from the literature.

The dual GP is equivalent to the traditional free energy minimization approach to the chemical equilibrium problem. The dual variables have a straightforward physical meaning: they are the number of moles of the species present at equilibrium. The disadvantages are the bigger dimensionality of the dual, the non-differentiability of the objective function at zero ,and the numerical problems that result when one variable tends to zero (Gochet _et_ _al_. , 1974).

The primal problem has less variables and few constraints. The constraints are highly non-linear, but they are inequality ones, and they are active at the optimum. Two related problems remain: the scaling of the problem and the finding of a first primal starting point.

We decided to try one computer code that could be used for both the primal and the dual formulations, in order to compare the two cases. GRG had been proven in the literature as comparable, if not better than most general-purpose GP codes. It could also be applied on the dual and the primal problems. A 1975 version of the code was available at UBC (Wales, 1977) and could be used as a self-contained program and as a subroutine. We decided to use GRG as our comparison code.

In this preliminary part of the work we were concerned with the accuracy of the results. We had to determine the best conditions for GRG to get results comparable to the literature. The general scaling of the problems was a difficult task, and three different versions of solving it were tried. All this was performed with GRG as a self-contained program.

The rest of the chapter is organized in the following way. In the first two sections we respectively repeat the chemical equilibrium formulations that we used, and give a detailed description of GRG. The third section deals with the scaling problem. The forth section states the working parameters for GRG. In the next section we present the method used to generate a starting point. The steps taken for performing sensitivity analysis are shown in the last section.

Mathematical formulations

For convenience we shall repeat here the mathematical formulations of the primal and the dual chemical equilibrium problems as we used them. They correspond to equations 2.40, 2.41, 2.42 and 2.43 of chapter II. For programming reasons, we incorporated the normality condition to the exponent matrix and to the B vector, and wrote the B vector explicitly. All subscripts are shifted one unit; A11=B1=1, Ai1=0 for all i, A1j=0 for all j>1

Dual problem

$$\text{Min } (-v) = \exp(G/RT) = \prod_{j=1}^{N+1} (c_j/\delta_j)^{\delta_j} \prod_{k=1}^{K} \lambda_k^{\lambda_k}$$

s. to

$$\sum_{j=1}^{N+1} A_{ij}\,\delta_j = B_i \qquad i = 1,M+1$$

$$\delta_j \geqslant 0 \qquad j = 1,N+1 \qquad 4.1$$

Transformed dual

$$\text{Min } (-\ln v) = G/RT = \sum_{j=1}^{N+1} \delta_j (\ln \delta_j + Cj) - \sum_{k=1}^{K} \lambda_K \ln \lambda_K$$

s.to

$$\sum_{j=1}^{N+1} Aij\ \delta_j = Bi \qquad i = 1, M+1$$

$$\delta_j \geqslant 0 \qquad j = 1, N+1 \qquad 4.2$$

Primal problem

$$\text{Min } g_o(t) = \prod_{i=1}^{M} t_i^{-Bi+1}$$

s. to

$$g_K(t) = \sum_{j \epsilon k} c_j \prod_{i=1}^{M} t_i^{Ai+1,j} \leqslant 1 \qquad k = 1, K$$

$$t_i \geqslant 0 \qquad i = 1, M \qquad 4.3$$

Transformed primal

$$\text{Min } \ln g_0(t) = h(Z) = \sum_{i=1}^{M} -B_{i+1} Z_i$$

s. to

$$\sum_{j \in k} \exp(C_j + \sum_{i=1}^{M} A_{i+1,j} Z_i) \leq 1 \quad k = 1,K \qquad 4.4$$

GRG. Description of the code

The program for the GRG code available at UBC was written at Cleveland State University. The manual UBC GRG, written by K. Wales (1977) is largely taken from Cleveland State University technical memorandum C1S-75-02 (1975). We shall repeat here some of the more important features of this code.

Stating the problem with GRG

Warning: the nomenclature used by GRG clashes with ours. Later in this chapter we include a table to compare them (Table IV-3)

GRG solves constrained optimization problems stated as follows:

$$\boxed{\begin{array}{lll} \text{Min } g_{M+1}(x) & & \\ \text{s. to} & & \\ & g_i(\underline{x}) = 0 & i = 1,NEQ \\ & 0 \leqslant g_i \leqslant UB_{N+i}\ ) & i = NEQ+1,\ M \\ & LBi \leqslant \underline{x} \leqslant UBi & i = 1,N \end{array}} \qquad 4.5$$

Where:

$\underline{x} = (x_1, \ldots x_N)$ vector of N real variables.

$\underline{G} = (g_1, \ldots g_{M+1})$ vector of real continuous functions of $\underline{x}$, linear or non-linear.

$g_1 \ldots g_{NEQ}$ equality constraints.

$g_{NEQ+1} \ldots g_M$ inequality constraints.

g_{M+1} objective function.

LBi lower bound on x_i $i = 1, .. N$

UBi upper bound on x_i $i = 1, .. N$

UB upper bound on the inequality constraint g_i $i = NEQ + 1, M$

<u>Description of the algorithm</u>

We will now give a brief description of the algorithm.

M slack variables are added to the constraints of problem (4.5). The previous N variables are called natural variables. Assume $\underline{x}$ is a feasible point, and NB of the constraints are binding at $\underline{x}$. Two sets of variables are distinguished in the GRG algorithm, provided there is no degeneracy : the NB basic variables (dependent) and the non-basic variables which are the N-NB remaining natural variables and the M slack variables associated with the binding constraints.

The binding constraints can be written as

$$G\ (Y,Z) = 0 \qquad 4.6$$

where Y is the vector of the NB basic variables and Z is the vector of non-basic variables. Then the binding constraints may be solved for Y in terms of Z, yielding a function Y(Z), valid for all Y,Z close to the first feasible point. This reduces the objective function to a function of only the Z variables, F(Z) which is called the reduced objective. The gradient of F(Z) is called the reduced gradient. The original problem is now a reduced problem.

$$\begin{aligned} &\text{Min } F\ (Z) \\ &\text{s. to} \\ &\qquad LB \leqslant Z \leqslant UB \qquad 4.7 \end{aligned}$$

Where LB and UB are respectively the lower and upper bounds for Z, and the vanishing of the reduced gradient is sought.

GRG solves (4.5) (with the slack variables) by solving a sequence of (4.7) reduced problems. Each problem is solved by a gradient type unconstrained non-linear optimization method. The reduced gradient gives a search direction d for a one dimensional linear subsearch as to minimize $F(Z+ \alpha d)$ with respect to α . A set of final equations is solved using Newton's method.

If Newton does not converge, GRG reduces α and tries again. Otherwise the search is finished. If Newton converges, but a basic variable violates the bounds, a new set of basic variables is determined and the solution of a new problem starts. The search may continue until the objective is found larger than the one in the previous iteration. Then a quadratic interpolation is done to the three α values bracketting the minimum, and the objective is evaluated at this point.

If the initial $\underline{x}$ does not satisfy the constraints, GRG optimizes a "Phase 1" objective which is the sum of the constraints violations, in order to obtain a first feasible point.

Use of GRG

GRG can be used as a self-contained program or as a subroutine. The self-contained program was used in the preliminary stages of this work. Details on the use of GRG can be taken from the UBC-GRG writeup (Wales, 1977).

Some parameters have to be specified for GRG to work. The values of these parameters depend on the mathematical formulation of the problem, on the scaling of the problem, and on the accuracy required for the solution, and will be discussed later. Running GRG as a self-contained program , these parameters are prompted by the user in a terminal on conversational mode. The objective function and the constraints are calculated by a subroutine named GCOMP, provided by the user in a file attached to input unit 5.

The scaling problem

The fact that the chemical equilibrium is an ill-conditioned problem was pointed out by many authors (Beightler and Phillips, 1976; Dembo, 1976; Lidor, 1975). One difficulty is the huge value of the GP objective function; the range of the primal variables at the optimum is quite wide. The free energy coefficients also vary a lot between species.

Table IV-1 exemplifies the need for a scaling with some problems from the literature.

All the figures in table IV-1 are from the literature or they were calculated from literature data.

Problem 1 was solved in the literature (Passy and Wilde, 1968) as a primal problem. The t variables were scaled so that their values at the optimum had a range of 100 between the lower and the higher values. No comments were made in the literature on the logic of the scaling.

Problem 4 is one of Dembo's test problems for evaluating geometric programming codes. GRG could not solve the unscaled version when using the t variables. The problem was scaled by Dembo, and its results were reproduced by us, starting from a point closer to the optimum than theirs. The transformed primal problem could be solved unscaled.

The sources of problems 2,6 and 7 did not specify the value of the objective function, which we calculated from their composition and free energy data. Problem 5 was solved in the literature as a dual GP, using the logarithm of the objective function.

As a first approach, we tried to reproduce the problems from table IV-1 using GRG. We succeeded in the scaled and transformed problems, but not on the unscaled versions of the primal or dual geometric programs. See appendix B for our results.

Attempts to solve the scaling problem

From what we could see in the literature, the attempts done to scale the primal problem were:

a) when working with the primal variables t, the objective function was elevated to an exponent so as to diminish its absolute value, and the primal variables were scaled in obscure ways so that they would fall within a close range from their optimum value.

b) when working with the transformed primal variables Z, there was no scaling problem. The objective function corresponded to the standard Gibbs free energy of the system and the transformed primal problem was a convex program. As can be observed in table IV-1, the range of variation of the transformed primal variables is not too wide.

We tried both schemes a) and b) on problems 5 and 6, running

Table IV-1. Characteristics of the Chemical Equilibrium Problem[1]

PROBLEM NUMBER	SPECIES, ELEMENTS, PHASES.	OBJECTIVE FUNCTION @ OPTIMUM			RANGE OF VARIABLES @ OPTIMUM		
		v*	$v^{.01}$	ln v	t**	Z†	δ‡
1	10, 3, 1	6.2904 E20	1.61431	47.8907	5.61 E-5 2.47 E-7	-9.7882 -15.214	1.48 E-1 6.93 E-4
2	4, 3, 1	1.9699 E39	2.47141	90.4788	4.93 E-2 2.86 E-14	-5.3120 -33.488	2.48 E-1 2.52 E-1
3	5, 3, 1	2.9231 E34	2.21136	79.3605	4.25 E0 1.49 E-10	-.85543 -24.927	1.72 E-1 5.79 E0
4	30, 12, 3	7.8257 E796	9.30969 E7	1834.91	6.51 E-1 1.19 E-9	-.42960 -20.552	6.60 E-21 28.9 E1
5	10, 4, 1	5.2990 E377	2.38365 E3	777.638	1.10 E-5 1.80 E-9	-11.415 -20.127	4.40 E-4 19.9 E1
6	8, 4, 1	1.4576 E52	3.3281	120.111	5.84 E-3 3.32 E-20	-5.1423 -44.851	1.41 E-4 1.88 E0
7	24, 4, 1	1.4448 E52	3.32352	120.102	6.07 E-3 3.32 E-20	-5.1037 -44.852	5.77 E-22 1.88 E0

----
[1] The source data for this table are in Appendix B
* v = exp (G/RT)
** t = primal variables
† Z = lnt
‡ δ = dual variables (number of moles)

GRG as a self-contained program. When using the primal variables t , the objective function was elevated to 0.01 and the primal variables t were scaled based on a primal starting point close to the optimum. After the scaling, the variables were arbitrarily bounded between 0 and 500.

When using the transformed primal variables Z, we were faced with the problem of fixing the boundaries for these variables.

We put zero as upper bound for all the variables. The tranformed primal objective function is linear on the Z variables; each Z can be regarded as the contribution of the associated element to the total free energy of the system (Zeleznik and Gordon, 1968). Therefore, if Zi is equal to zero that means that the element i is not contributing to the total free energy of the system, regardless of the amount of element i. This is quite an unrealistic situation if the model is well posed, and so zero looks like an appropiate upper bound for the Z variables.

The lower bound is not so easily determined. A fixed large negative number like -100 is one possibility, but this approach has some problems: it does not account for cases when Z is below that boundary and the range may be too wide for some variables. Also, in the evaluation of the terms of the primal constraints we can have exponentials of too large negative numbers, which are undefined.

A second possibility to determine general lower bounds for the transformed primal variables is to calculate a fixed percentage of a first good starting point and use it as a boundary. But then there is the question of how big this percentage should be; we tried a few numbers, and found variations from problem to problem.

The third possibility is to add a fixed negative number to a good starting point. If we choose -30, that is equivalent of a range of 10 ^{13} in the untransformed primal variables t. If we choose -20, the range is 10 ^{8}. We chose -30 arbitrarily and it worked well in all our examples . This selection of a boundary is then strongly dependent on the first starting point. Using an aproximation of the problem to a linear program allowed us to obtain dual starting points that are quite close to the optimal values. The corresponding primal points can be calculated from the dual. A description of the procedure is done later in this chapter. See Chapter VI for the results of the closeness of these points to the optimum values.

Comparison between scaling the primal and using the transformed primal

Table IV-2 shows the results of a comparison between the two procedures explained in the previous section. The problems No. 5 and 6 of table IV-1 were scaled, and also posed as transformed primal problems. Both problems have 4 primal variables, and according to the literature (Gochet et al. , 1974) for general GP the problem posed as t variables should be solved more efficiently by GRG than the transformed one. The number of iterations and of function evaluations needed to go from the same starting point to the same optimum were computed with both

Table IV-2. Performances of the scaled primal problem and the ln-transformed primal with GRG (*)

Problem	Scaled		Ln-transformed	
	Iterations	F. evaluations	Iterations	F. evaluations
5 (**)	7	51	5	46
6 (***)	20	240	13	183

Notes:

(*) GRG parameters: EPSTOP=EPNEWT=EPSBOUND=ESPIV= 10^{-6}

(**) Scaling: objective function elevated to .01 ; y1=t1.10^5
y2 = t2. 10^6 , y3 = t3 . 10^6 , y4 = t4 . 10^5
Starting point : Z1=-11.42, Z2=-20.13, Z3=-15.86, Z4=-11.63
Optimum : Z1=-11.4315, Z2=-20.13541, Z3=-15.8013, Z4=-11.6964

(***) Scaling: objective function elevated to .01 ; y1=t1.10
y2=t2. 10^{23} , y3=t3. 10^7 , y4=t4 . 10^5
Starting point: Z1=-4.35, Z2=-51.6, Z3=-16.3, Z4=-11.9
Optimum: Z1=-4.3370, Z2=-52.2190, Z3=-15.9968, Z4=-11.9352

methods. The same tolerances were used in all cases.

It took GRG less iterations and function evaluations to solve the transformed primal, the opposite of what the literature said for general GP . We believe this is due to the particular ill-conditioning of the chemical equilibrium problem, and therefore should need a special scaling for each example. From here on, we will use the words "primal" and "transformed primal" indistinctly, and we will always refer to the ln-transformed problem.

For the dual problem, we found no special computing advantages for either using the objective function as in a standard geometric program (but elevated to a certain exponent), or minimizing the negative logarithm of that function. The second approach was attractive because then the objective function represented the Gibbs free energy of the system divided by R and by T, same as in the transformed primal, and as in the traditional "free energy minimization method". For the rest of the thesis , whenever "dual problem" is mentioned, we will be referring to equations 4.2.

Determination of working parameters for GRG

GRG needs as an input a series of control cards that have to be provided by the user. We determined some of them from literature values; some others through numerical experience. The resulting values may not be the best for a specific problem; however, they allowed us to obtain results similar to the literature in all cases. Table IV-3 explains the meaning of the key words and exemplifies the values used in our programs.

A dual starting point

A starting point for the dual problem is quite straightforward, since the dual variables are the number of moles of the chemical species. When using GRG as a self-contained program, we usually started with a point close to the literature optimum, and then we changed it at random.

For the user's convenience, the final code should include the generation of a starting point. The NASA code uses an equal

TABLE IV-3. Parameters for GRG

GRG PARAMETERS	Transformed PRIMAL	DUAL
$N \leq 100$ (Number of variables)	M + 1 (Number of elements + 1)	N + 1 (Number of species + 1)
$M \leq 100$ (number of constraints)	K (number of phases)	M + 1 (number of elements + 1)
NEQ (number of equality constraints)	0	N + 1 (number of elements + 1)
LBV (lower bound on variables	Starting point +(-30)	E-24 (E-10 - E-30) -tried-
UBV (upper bound on variables)	0	40
UBC (upper bound on inequality constraint)	1	0
X (initial vector of variables)	Generated with LIPSU2 and SINGV	Generated with LIPSU2
EPNEWT (tolerance for equality and binding constraints)	E-6 (very important for accuracy)	E-6 (very important for accuracy)
EPSBOUND (tolerance for variables @ bounds)	E-6	E-6

//continues

///Table IV-3 (continued)

GRG PARAMETERS	Transformed PRIMAL	DUAL
EPSTOP (tolerance for objective function stopping criteria)	E-6	E-6
EPSPIV (tolerance for pivot element in the basis)	E-6	E-6
QUAD (quadratic extrapolation for estimating initial basic variables)	used -important-	used -important-
PRINTCTL (control amount of output)	2 for self-contained program; 1 for subroutine	2 for self-contained program; 1 for subroutine

distribution of species for that purpose. The RAND code approximates the dual problem to a linear program to obtain a first point. Lidor (1975), did a different approximation than the NASA , and obtained another linear program for his starting point routine. Lidor compared his results to those of the RAND code, and got points closer to the optimum. We decided to implement Lidor's approach.

His formulation of the linear program (LP) is as follows:

<u>LP approximation of the dual</u>

$$\text{Min } f = \sum_{j=1}^{N} Cj\, y_j + \xi \sum_{j=1}^{N} Cj$$

s. to

$$\sum_{j=1}^{N} Aij\, y_j + \xi = Bi \qquad i = 1,M$$

$$\xi > \epsilon$$

$$y_j \geqslant 0 \qquad j = 1,N \qquad 4.8$$

Then:

$$\delta_j = y_j + \xi \qquad 4.9$$

To solve the linear program, we used subroutine LIPSUB from UBC (Patterson, 1979) on the program LIPSU2 . The following steps were followed:

1) Read C, A and B.

2 Set the tableau in the manner specified by the UBC routine LIPSUB .

3) Call LIPSUB . It solves the linear program using a primal-dual algorithm.

4) The results (the y variables and ξ) are used to calculate the dual starting point with equation 4.9.

Calculation of primal variables from the dual ones

The primal starting point is more difficult to visualize and is more critical than the dual. We already mentioned that we are defining the lower bounds of the primal variables as a certain function of their starting value. Since we already have a program to generate a dual starting point, it sounds reasonable to transform this point to primal variables.

For a geometric program at the optimum , the primal and dual variables are related as follows:

$$\sum_{i=1}^{M} Aij\ Zi = Cj + \ln(\delta_j/\lambda_k) \qquad j = 1,N \qquad 4.10$$

where

$$\lambda_K = \Sigma_{j\in K}\delta_j \qquad k = 1,K$$

There are M variables and N equations. In reacting chemical equilibrium problems, N>M and we have an overdetermined system of linear equations. The equalities hold only at the optimum;

this is a nuisance if we are interested in the starting point which is not normally the optimum.

A linear least squares approach should be the best solution. We tried subroutine DBEST from UBC with very good results. It decomposes the transpose of our A matrix (the exponent matrix) following a Gram-Schmidt orthogonalization. The problem was that the UBC subroutine was limited to cases with less than 30 species, so we tried another method.

Subroutine DSLVD from UBC (UBC MATRIX, 1980) uses a singular value decomposition to solve overdetermined systems of linear equations. The transpose of the matrix A is decomposed as follows:

$$A' = U \Sigma V' \qquad 4.11$$

where

U, V = orthogonal matrices

Σ =diagonal matrix of the singular values of A'

The solution of the system of equations $A'Z = b$ is the vector Z

$$Z = V \Sigma^{+} U'b \qquad 4.12$$

Where Σ^{+} is the pseudoinverse of Σ.

We implemented the program SINGV as follows:

1) The exponent matrix A, the free energy coefficients matrix C, the dual starting point and the values of M and N are read.

2) Call subroutine DLSVD from UBC. It calculates the matrices V, Σ and U'b.

3) Calculate Σ^+ and V Σ^+.

3) subroutine DGMULT from UBC is called to multiply V Σ^+ and U'b and obtain the vector Z, which is actually the vector of the transformed primal variables z.

Sensitivity analysis

As we already discussed elsewhere (chapter II), there are two approaches in the literature to sensitivity analysis in geometric programming. Both approaches are mathematically equivalent but the first involves the inversion of a matrix and the second solves a system of linear equations.

We chose the combustion of propane (problem 5) as a means of comparison between the two approaches, because :

a) It was used by Dinkel and Lakshmanan in their paper on sensitivity analysis for the chemical equilibrium problem. They used the method of the inversion of the Jacobian (Dinkel and Lakshmanan, 1977).

b) it is a "middle size" problem within the literature. It involves six simultaneous reactions ; the dimension of the matrix to be inverted in the Jacobian approach is DxD, where D is the number of independent reactions . When D= 1 or 2, there is no doubt that this method will be faster than solving an equivalent system of N+K+1 by N+K+1 linear equations, where N= number of species, K is the number of phases. We showed before (Chapter II) that D=N-M; if we assume N=3, M=2, K=1, we have a 5 x 5 system of equations to be solved vs. elevating a number to -1. In the combustion of propane example, D=6, N=10, M=4, K=1 and then the comparison is between a 6 x 6 matrix to be inverted, or a 12 x 12 system of linear equations to be solved.

For the purpose of the comparison the stoichiometric coefficients used were the ones calculated by Dinkel and Lakshmanan (1977). We will now explain a systematic procedure to obtain these coefficients from the exponents matrix AA.

Calculation of stoichiometric coefficients

A systematic method to construct a maximal set of linearly independent chemical reactions from the exponent's matrix AA is as follows: (Cavallotti |et al , 1979)

a) Given the matrix AA, factorize it into two matrices E and F, where E is an MxM matrix of rank M, and F has dimensions MxD, so that, in matritial notation:

AA = | E | F |

In chemical language, that is equivalent to reaccommodating the matrix AA, so that the first M columns correspond to the key components; the matrix F is the matrix of the constituents.

b) Reduce the factorized matrix by a Gauss Jordan procedure until it becomes

AA = | I | H |

where I is the MxM identity matrix.

c) Form the array :

$$U = \begin{vmatrix} -H \\ I \end{vmatrix}$$

Where I is now the DxD identity matrix, and U stands for the NxD matrix of stoichiometric coefficients for the D reactions. With this construction, the product of matrices AA.U = 0 , which was the condition required for the stoichiometric matrix (see chapter II).

<u>Comparison between the two methods of sensitivity analysis</u>

We implemented Dinkel and Lakshmanan's method in the program JOTA (appendix B). The method based on the solution of the linear system of equations is contained in program NPLUSK (appendix B). We will now describe both algorithms.

<u>a) Algorithm of program JOTA</u>

1) read optimal composition at optimum temperature and pressure (data from the literature), stoichiometric coefficients, total number of moles in the phase, free energy coefficients at the optimal and at the perturbed T,P.

2) form the J matrix according to equations 2.44, 2.45, 2.46, 2.47.

3) the J matrix is inverted by the UBC subroutine INV (UBC Matrix, 1979). The execution time to invert the matrix is printed.

4) for the new set of free energy coefficients, the new composition and free energy are computed using equations 2.49

and 2.48, and these values are printed.

B) Algorithm of program NPLUSK

1) read same data as in JOTA .

2) the matrix to be solved is posed at the optimal conditions of P, T, and composition. The equations 2.50 to 2.53 are used for this purpose.

3) the right-hand side vector is also calculated for the perturbed condition.

4) the system of equations is solved by UBC subroutine DSLIMP (see UBC MATRIX, 1979). The execution time for the performance of DSLIMP is printed.

5) the new composition values are printed.

In table IV-4, we can see that both programs yield figures for composition comparable to those obtained in the literature for sensitivity analysis, but the execution time for the JOTA program was a thousand times longer. As a consequence of this, we decided to incorporate the NPLUSK program as a subroutine of our main program in order to perform sensitivity analysis (see Chapter 5).

When the free energy coefficients of the species change, only the right hand side of the system of equations needs to be recalculated. Since the matrix of the equations is solved by DSLIMP, it is very easy to make a new perturbation and resolve the system, for the matrix is stored . It is not the same when the amounts of elements are changed. In this case, the first column of the matrix has to be recalculated. No attempts were

Table IV-4. Two approaches to sensitivity analysis. Composition values for problem 5 * , at two thermodynamic conditions, using optimization. Composition at the perturbed condition, calculated with two methods of S.A. Execution times for the two methods.

Species	Start T=2200 K P=40 at.	Perturbation : T = 2500 K , P = 50 at.			
		Optimiz.*	S.A. *	JOTA ** t=2.24008 s	NPLUSK *** t=0.00161 s
H2	.02007	.05492	.04020	.04020	.04017
H	.00065	.00433	.00189	.00189	.00189
OH	.01500	.05579	.03534	.03534	.03532
H2O	3.9719	3.9150	3.9412	3.9412	3.9412
CO	.08160	.24835	.17378	.17379	.17267
CO2	2.9184	2.7516	2.8262	2.8262	2.8263
N2	19.987	19.959	19.971	19.972	19.972
NO	.02668	.08618	.05680	.05679	.05679
O2	.03358	.09614	.06954	.06954	.06955
O	.00044	.00356	.00137	.00139	.00138

T= temperature, K= Kelvin, P= pressure, at.= atmospheres,
SA.= sensitivity analysis, Optimiz.= optimization,
t= execution time, s= seconds.
* Dinkel and Lakshmanan, 1977.
** JOTA : S.A. program based on inverting the Jacobian.
*** NPLUSK : S.A. program that solves a system of linear equations.

done on solving this case.

From the data in table IV-4 we can see that the relative errors of the composition calculated by sensitivity analysis, are quite large. These errors are relative to the value of the composition as calculated by a new optimization. The errors also vary from species to species.

Almost all papers on sensitivity analysis in GP (Rijckaert, 1974; Dinkel and Lakshmanan, 1975,1977) propose an incremental procedure to diminish these errors. An evaluation of the size of the increment was done on the example No. 5, and is explained in Chapter VI.

Sensitivity analysis seems a faster way to solve the chemical equilibrium problem in a region close to a known optimun, than reoptimization itself. Just how much faster it can be, with similar accuracy, has not yet been stated. We tried to compare the sensitivity analysis procedure vs. reoptimization, and the results and conclusions are in Chapter VI.

Conclusions of this chapter

The conclusions of these preliminary studies are:

1) GRG solves both primal and dual GP chemical equilibrium problems, provided there is a scaling, or a logarithmic transformation. It is more effective if (i) the primal is posed as a problem of the transformed primal variables $Z= \ln t$, (ii) the logarithm of the dual objective function is optimized.

2) The working parameters for GRG were stated.

3) A program to generate dual starting points was written.

4) A program to calculate primal points from the dual ones was implemented.

5) Sensitivity analysis for the variations of the free energy coefficients on the dual variables was performed faster by solving a system of linear equations rather than by inverting the Hessian matrix. The problem treated had a degree of difficulty (number of independent reactions) equal to six.

6) For sensitivity analysis we need the stoichiometric coefficients for the reactions. A literature based systematic procedure to obtain them from the exponents matrix was

explained.

CHAPTER V: DESCRIPTION OF THE PROGRAMS

Diagrams

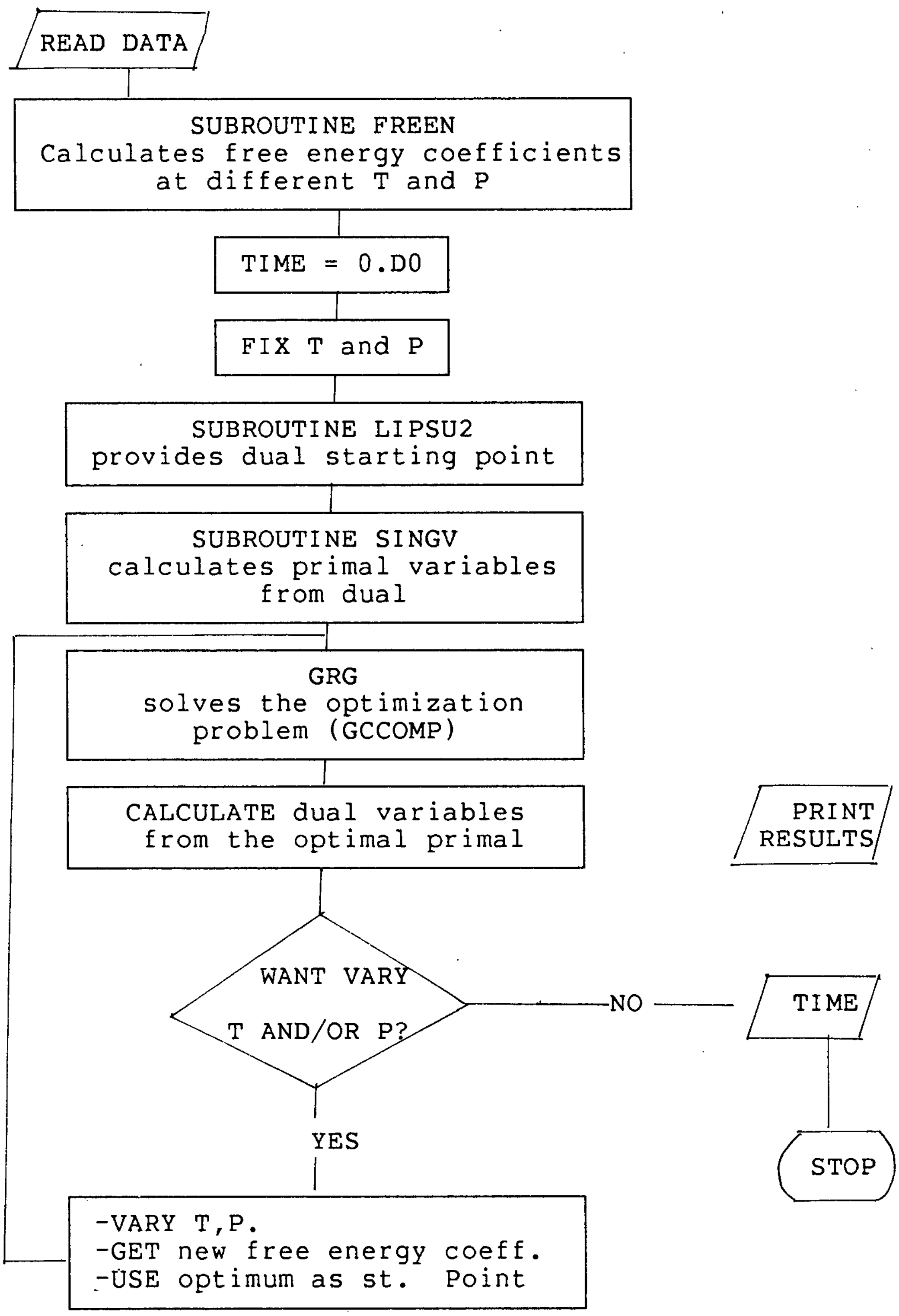
We will present here the diagrams of the four computer programs that were written. The idea was to compare primal vs. dual perfomance, and sensitivity analysis vs. re-optimization. The diagrams are quite rough; a more detailed explanation of each subroutine will be given in the following sections of this chapter.

Program COMP 1 solves the primal transformed problem with GRG. Program COMP 2 solves also the primal, and performs sensitivity analysis as well. Program COMP 3 solves the dual and finally program COMP 4 solves the dual and performs sensitivity analysis.

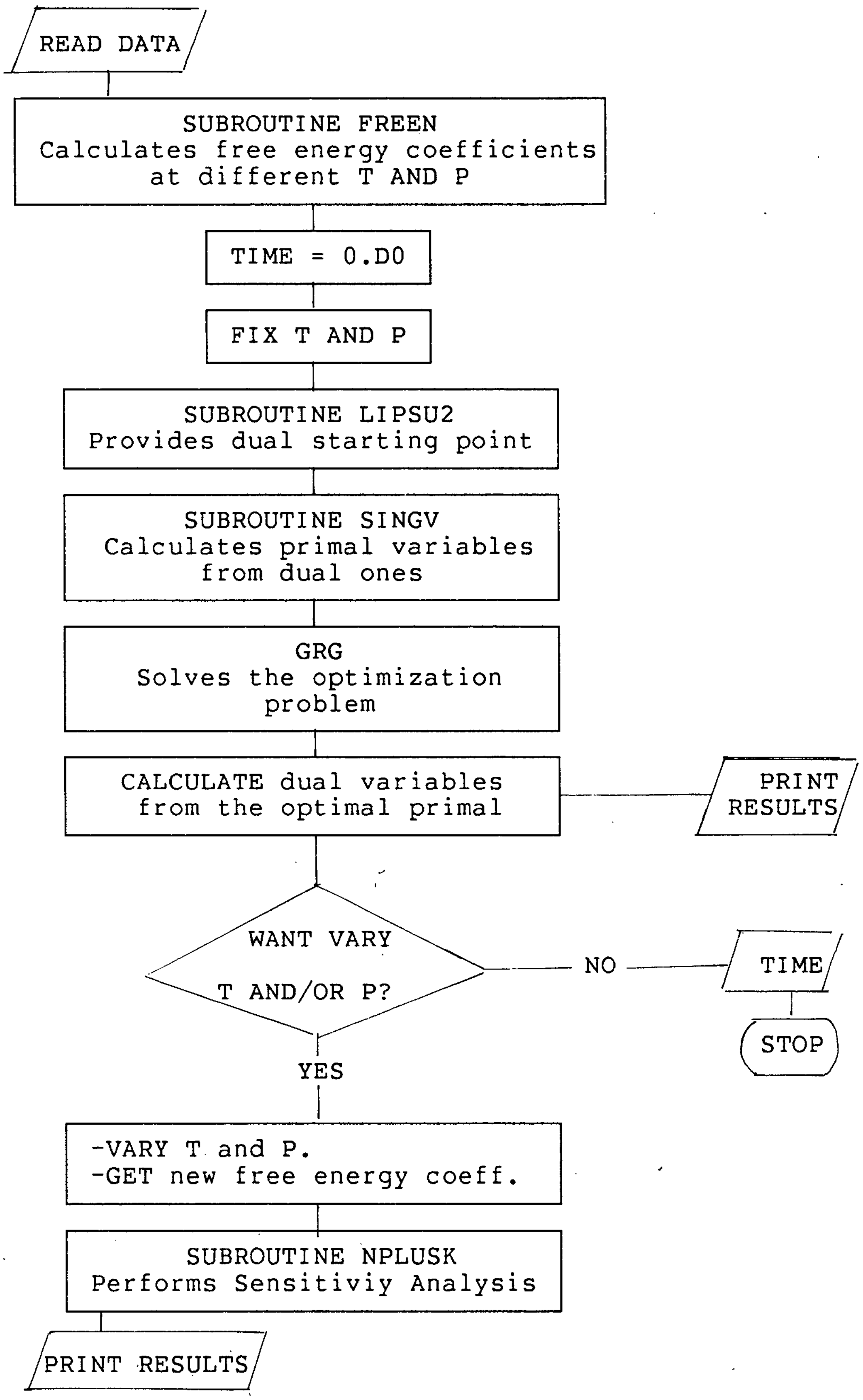
The programs are written in FORTRAN IV . All subroutines are related through labelled COMMON blocks. Each subroutine calls at least one subroutine from UBC (except for subroutine FREEN).

The programs were run on an AMDAHL 470 V/8 computer, using the Michigan Terminal System. They were compiled with IBM FORTRAN compilers G and H at level 21.8. Data were read from a file attached to input unit 5.

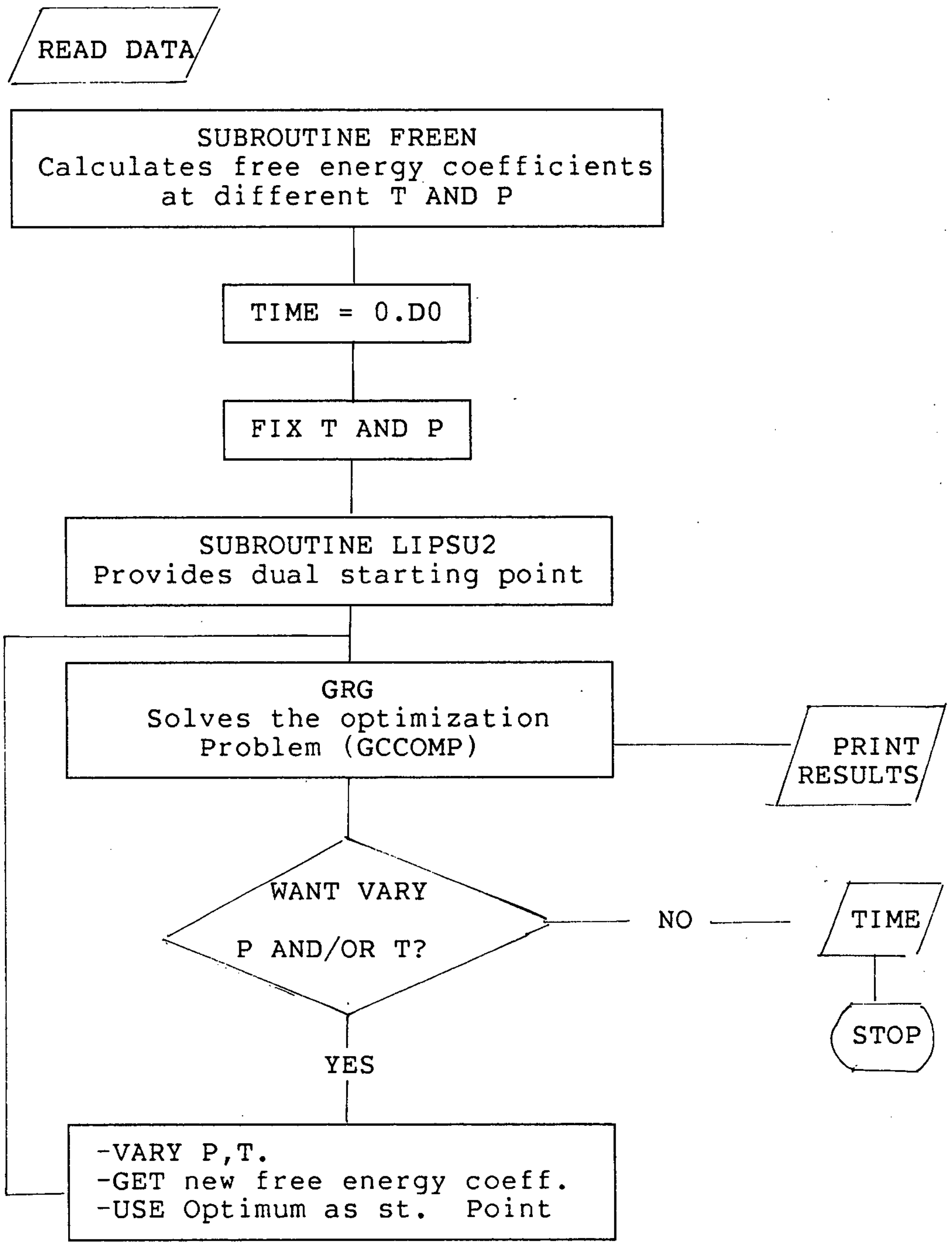
PROGRAM COMP1


READ DATA
SUBROUTINE FREEN
Calculates free energy coefficients at different T and P
TIME = 0.D0
FIX T and P
SUBROUTINE LIPSU2
provides dual starting point
SUBROUTINE SINGV
calculates primal variables from dual
GRG
solves the optimization problem (GCCOMP)
CALCULATE dual variables from the optimal primal
PRINT RESULTS
WANT VARY T AND/OR P?
NO
TIME
YES
STOP
-VARY T,P.
-GET new free energy coeff.
-USE optimum as st. Point

PROGRAM COMP2

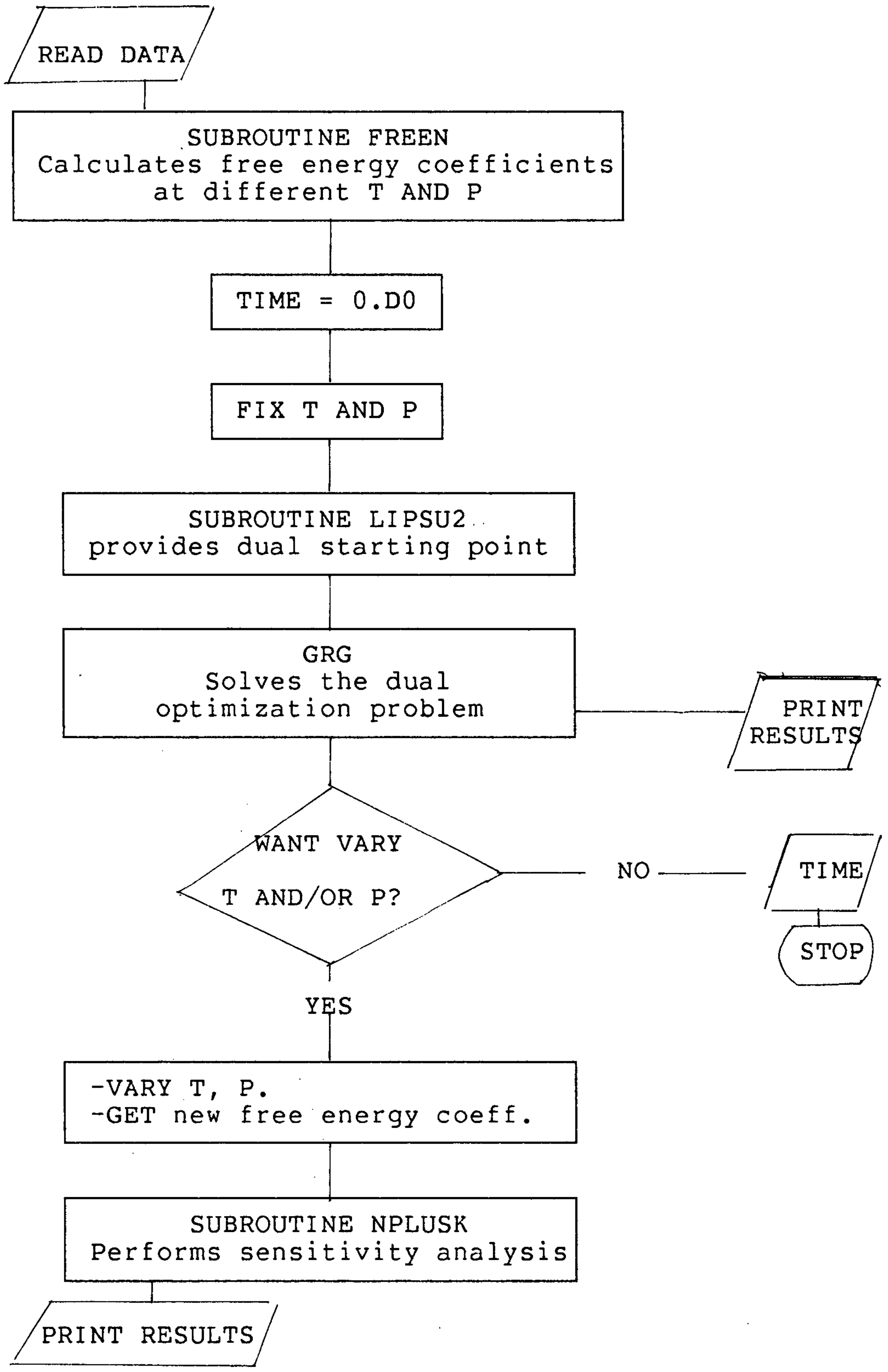

READ DATA
SUBROUTINE FREEN
Calculates free energy coefficients at different T AND P
TIME = 0.D0
FIX T AND P
SUBROUTINE LIPSU2
Provides dual starting point
SUBROUTINE SINGV
Calculates primal variables from dual ones
GRG
Solves the optimization problem
CALCULATE dual variables from the optimal primal
PRINT RESULTS
WANT VARY T AND/OR P?
NO
TIME
STOP
YES
-VARY T and P.
-GET new free energy coeff.
SUBROUTINE NPLUSK
Performs Sensitiviy Analysis
PRINT RESULTS

PROGRAM COMP3



READ DATA
SUBROUTINE FREEN
Calculates free energy coefficients at different T AND P
TIME = 0.D0
FIX T AND P
SUBROUTINE LIPSU2
Provides dual starting point
GRG
Solves the optimization Problem (GCCOMP)
PRINT RESULTS
WANT VARY P AND/OR T?
NO
TIME
STOP
YES
-VARY P,T.
-GET new free energy coeff.
-USE Optimum as st. Point

PROGRAM COMP4


READ DATA
SUBROUTINE FREEN
Calculates free energy coefficients at different T AND P
TIME = 0.D0
FIX T AND P
SUBROUTINE LIPSU2
provides dual starting point
GRG
Solves the dual optimization problem
PRINT RESULTS
WANT VARY T AND/OR P?
NO
TIME
STOP
YES
-VARY T, P.
-GET new free energy coeff.
SUBROUTINE NPLUSK
Performs sensitivity analysis
PRINT RESULTS

Input needed. Examples

For any of the programs mentioned above, the user should provide:

a) control cards:

N+1= number of species present at equilibrium +1

M+1= number of elements +1

K= number of phases

NN= number of times P and/or T are varied.

NE= 1 if the free energy coefficients are available as such; if they have to be calculated from the Gordon and Mc. Bride coefficients, use any other integer.

NF=1 if the dual starting point routine is to be used. Otherwise, use another integer.

b) free energy data: free energy coefficients matrix C or temperature coefficients matrix S for the Gordon and Mc. Bride polynomials, NASA SP-3001.

c) exponent matrix A. An example of its construction will follow.

d) amount of elements :vector B.

e) stoichiometric coefficients for the reactions if COMP2 or COMP4 are to be used (matrix U)

e) dual starting point if available.

We shall now give two examples of the input.

Example 1. CH4-Water gas reaction

Preliminary data:

-Reaction of 2 moles of CH4 and 3 moles of water

-Reach equilibrium at T= 1000K, P= 1atm.

-Species present at equilibrium = CO, CO2, H2O, H2,CH4

-Ideality assumed

-One phase (gas)

-Free energy coefficients available from the literature for one set of T,P conditions.

-Do not have a starting point.

Input

a) Control cards:

N+1 = 6 (5 species +1)

M+1 = 4 (3 elements +1)

K = 1 (1 phase)

NN = 1 (one thermodynamic state)

NE = 1 (the free energy coefficients are data)

NF = 1 (no starting point is provided)

b) Free energy coefficients :

C is a 6x1 matrix, as follows:

$$C = \begin{array}{rl} 0.0 & \text{dummy} \\ -24.025 & \text{CO} \\ -47.413 & \text{CO2} \\ -23.067 & \text{H2O} \\ 0.0 & \text{H2} \\ 2.0847 & \text{CH4} \end{array}$$

The first coefficient corresponds to the normality condition. The rest of the coefficients are the Gibbs free energy of formation from its elements at temperature = 1000K for each species, divided by R (the universal gas constant) and by the temperature, and added to the logarithm of the pressure (1 atm.), since there is only gas phase.

c) Exponents matrix :

For computing purposes , we added the normality condition to the exponents matrix AA described in the previous chapter. Hence the new matrix A is a (4 x 6) matrix constructed in the following way:

A (1, 1) = 1.DO

A (1, j) = 0.DO   j= 2,N+1

A (i, 1) = 0.DO   i= 2,M+1

the remaining of each column accounts for the formula of each chemical species. The final matrix looks like this:

$$A= \begin{array}{cccccccc} \text{dummy} & \text{CO} & \text{CO2} & \text{H2O} & \text{H2} & \text{CH4} & \\ 1 & 0 & 0 & 0 & 0 & 0 & \text{dummy} \\ 0 & 1 & 1 & 0 & 0 & 1 & \text{C} \\ 0 & 0 & 0 & 2 & 2 & 4 & \text{H} \\ 0 & 1 & 2 & 1 & 0 & 0 & \text{O} \end{array}$$

d) Vector of amount of elements B :

Since we started with 2 moles of CH4 and 3 moles of H2O, 2 atom-grams of C, 14 of H and 3 of O should be conserved throughout the reactions. Hence the B vector will be:

$$B = \begin{array}{cc} 1 & \text{dummy} \\ 2 & C \\ 14 & H \\ 3 & O \end{array}$$

e) since we only have one set of thermodynamic data, we are not concerned with sensitivity analysis and COMP2 and COMP4 should not be applied.

f) we do not have a starting point to enter here. The programs will generate one.

g) if we want to consider the formation of solid C, then the control cards and the exponents matrix should be modified as follows:

K=2 . For k=1, j varies between 1 and 6; for k=2, j=7

$$A = \begin{array}{cccccccc} \text{Dummy} & CO & CO2 & H2O & H2 & CH4 & C & \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & \text{Dummy} \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & C \\ 0 & 0 & 0 & 2 & 2 & 4 & 0 & H \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 & O \end{array}$$

The B vector remains as before. The C matrix has to incorporate the free energy coefficient for carbon (0) as C(7,1)

Example 2. Claus Furnace reaction

Preliminary data :

-Partially reacted mixture of 0.1 moles SO2, 0.2 moles H2S, 0.8 moles H2O and 1.88 moles of N2 (inert)

-Reach equilibrium composition at P = 1 atm., and 9 different T from 550 K to 1000 K (50 K intervals)

-Species present at equilibrium: SO2, H2S, H2O, S2, S4, S6, S8, N2.

-Ideality assumed.

-Elements : S, O, H, N

-One phase (gas)

-Temperature coefficients for thermodynamic functions of Gordon and Mc. Bride

-Want to compare sensitivity analysis versus re-optimization.

Input

a) Control cards:

N+1 = 9 (8 species)

M+1 = 5 (4 elements)

K = 1 (one phase)

NN = 9 (nine T,P conditions)

NE= 2 (free energy coefficients should be calculated)

NF =1 the starting point should be calculated within the program.

b) matrix of Gordon and Mc.Bride coefficients. Matrix S is an (N x 7) matrix each row of the matrix corresponds to the seven coefficients of the Gordon and Mc.Bride polynomials for a chemical species. The dummy species 1 is not included, so that the species here are shifted by one unit when compared to their numeration in matrix A. With these data the subroutine FREEN will calculate the matrix C.

c) Exponents matrix . Matrix A is as follows :

dummy	SO2	H2S	H2O	S2	S4	S6	S8	N2	
1	0	0	0	0	0	0	0	0	dummy
0	1	1	0	2	4	6	8	0	S
0	2	0	1	0	0	0	0	0	O
0	0	2	2	0	0	0	0	0	H
0	0	0	0	0	0	0	0	2	N

d) Vector of elements content . The B vector is as follows:

1.0	dummy
0.3	S
1.0	O
2.0	H
3.76	N

e) Matrix of stoichiometric coefficients: Since we want to apply sensitivity analysis , we need to provide the matrix of stoichiometric coefficients U . It is a (N+1 x D) matrix ; each column contains the stoichiometric coefficient of the species described by the row number for the D independent reactions. To the dummy species corresponds a stoichiometric coefficient equal to zero. If M+1 is the rank of the matrix A, then D=N - M. In this case D= 4 .

The U matrix is built in the following way :

-i) take matrix A, without the first row and column (they correspond to the normality condition of the geometric program). Exchange column 4 and 8, and partition the resulting matrix so that

$$\begin{array}{cccccccc}
1 & 1 & 0 & 0 & 4 & 6 & 8 & 2 \\
2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 0 & 0 & 0 & 0
\end{array}$$

-ii) work with elementary matrix operations to get

$$\begin{array}{cccccccc}
1 & 0 & 0 & 0 & 4/3 & 6/3 & 8/3 & 2/3 \\
0 & 1 & 0 & 0 & 8/3 & 12/3 & 16/3 & 4/3 \\
0 & 0 & 1 & 0 & -4/3 & -6/3 & -8/3 & -2/3 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{array}$$

-iii) form the stoichiometric matrix. Interchange the rows 4 and 8 to get the original set of species.

$$\begin{array}{cccc}
-4/3 & -6/3 & -8/3 & -2/3 \\
-8/3 & -12/3 & -16/3 & -4/3 \\
4/3 & 6/3 & 8/3 & 2/3 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0
\end{array}$$

To obtain the U matrix for our programs , a first row of zeroes should be added. If we multiply this U matrix by 3, the set of independent reactions is:

$$SO2 + 2\ H2S = 3/j\ Sj + H2O \qquad j = 2, 4, 6, 8.$$

Any other combination of reactions can be used, provided they are linearly independent.

Calculation of free energy coefficients

Subroutine FREEN calculates the free energy parameters of each species at a specified T and P. It uses the polynomial approximation to the Gibbs free energy determined by Gordon and Mc. Bride (1971). The polynomials are as follows:

$$\mu_j^{\circ}(T) / RT = S_{j1} (1 - \ln T) - S_{j2} T/2 - S_{j3} T^2 /6 - S_{j4} T^3 /12 - S_{j5} T^4 /20 + S_{j6} /T - S_{j7} \qquad 5.1$$

Where S is a Nx7 matrix of the Gordon and Mc. Bride coefficients for the N species. Then the free energy coefficients are calculated through the definitions:

$$Cjl = \mu_j^{\circ}(T) / RT + \ln P \qquad 5.2$$

for gases and

$$Cjl = Gj(T) / RT \qquad 5.3$$

for condensed phases.

The subscript l refers to the number of times T and P are varied.

All the information needed by the routine, and produced by it, is handled by means of labelled COMMON blocks. When other sources of free energy data are available, the programs skip subroutine FREEN and read the matrix C supplied by the user.

Subroutine LIPSU2 . A dual starting point.

If a good first guess of the composition is available, the programs may take this guess as a starting point. If this is not the case, the programs generate a dual starting point calling the subroutine LIPSU2 . This subroutine is almost identical to the program LIPSU2 explained in the previous chapter. The subroutine obtains the data from the main program through labelled COMMON blocks and it sets the tableau in the manner specified by the UBC routine LIPSUB . LIPSUB is then called and it solves the LP using a primal-dual algorithm. The results are stored in the COMMON block FX. The main program then uses this information to calculate the dual starting point.

Subroutine SINGV

Subroutine SINGV transforms dual variables into primal ones. It is very similar to the program of the same name described on chapter 4. The reading and output are replaced by labelled COMMON blocks.

Using GRG as a subroutine

GRG is called from the main program as three subroutines from UBC. GRGIN reads the control cards described in chapter IV. GRG2 calls the optimization subroutine. The value of the objective function and of the variables at the optimum are arguments of GRG2. GREG prints the Lagrange multipliers of the binding and equality constraints, and the characteristics of the optimization such as number of iterations, number of function evaluations, etc.

Giving input to GRG

To read the control cards needed for the optimization, subroutine GRGIN does it from a file attached to the logic unit 5. That introduces two problems :

a) The main program reads its own data from a datafile attached to the input unit 5.

b) Some of the parameters that should be specified in the control cards are calculated in the first part of the program, like the starting point and the boundaries. Hence the control cards cannot be settled in advance.

To solve these two points, a scratch file -DATA is created from the main program. Input unit 5 is reassigned to -DATA . The control cards are then written into -DATA using the information available in the main program , with the required format.

Subroutine GCOMP

GRG2 is now ready to perform the optimization. It needs to evaluate the objective function and left-hand sides of the constraints for the given values of the variables x . To do that, it calls subroutine GCOMP. The form of this subroutine will depend on the mathematical formulation of the problem we are dealing with, but the variables have to be called x. Subroutine GCOMP gets the needed data through labelled COMMON blocks.

For the dual case, the problem of logarithms with zero or negative arguments is avoided by defining the logarithmic function as equal to a very large negative number when negative or zero variables are encountered in the search. Also, the lower bound of the variables is set to a very low value. After the optimization is finished, any number of moles smaller than the inverse of the Avogadro's number is set to zero by the main program.

In the primal problem, the exponentiation of very large negative numbers may occur, and this is also undefined. We do not allow the lower boundaries of the primal variables to go below the defined region.

To delete species from the model, the corresponding column of the exponent matrix is set to zero. For the primal problem, that means that the molar fraction of the corresponding species is one. To avoid the problem, we defined DEXP(0)=0 when Aij=0.

Output from GRG

The amount of output produced by GRG is controlled by the keyword PRINTCTL=number which should be specified in a control card. When PRINTCTL=0, no information is output. When it is set equal to 1, the results are printed; when working at a terminal, the user is prompted to write the word GO after the initial conditions for each optimization are set. Succesive information on the progression of the optimization is printed when PRINTCTL values range from 2 to 4. The input lines are echoed by default. If this is not desired, NOECHO has to be specified in a control card.

The values of the objective function and of the variables at the optimum is all the information needed from GRG when solving the dual problem. They are arguments of GRG2 , hence for this case we can avoid any printed output from GRG if desired. When solving the primal, we need also the Lagrange multipliers for the binding constraints (they are the total number of moles for each constraint when solving the transformed primal problem), and they are not available as arguments. We had to get them through a rather twisted way.

The main program creates a scratch file -GRGOUT, and the output unit 6 is reassigned to this file. Subroutine GRGEG, from UBC, writes the output from the optimization into -GRGOUT, provided PRINTCTL is set to a value different than zero. Subroutine SKIP then scans through -GRGOUT until it finds the expression "Lagrange multipliers". At this point, the main program reads the values of the multipliers from the next line on the output scratch file, and calculates the number of moles for each species. -GRGOUT is rewound after each optimization. Listing the file allows us to compute the number of iterations and of function evaluations of the last run.

Performing sensitivity analysis

Subroutine NPLUSK is very similar to program NPLUSK described in chapter IV. The input of data is done through labelled COMMON blocks. The stoichiometric coefficients have to be provided by the user, since we did not implement a subroutine to calculate them from the exponents matrix, as exemplified earlier this chapter (section 5.2.)

Subroutine NPLUSK formulates the linear system of equations from the data available. The right hand side of the equations varies with pressure and temperature, and is recalculated each time these conditions change, but the left hand side remains the same. Subroutine DSLIMP from UBC solves the system of equations.

Output from the programs

If the keyword PRINTCTL is set equal to 1, the output of the programs contains :

a) Matrix of free energy coefficients (if applicable).

b) Dual starting point.

c) Primal starting point (if applicable).

d) Input for GRG

e) Pressure and temperature

f) Number of moles of the species present at equilibrium at the previous P and T.

g) Total number of moles per phase at these conditions.

h) Corresponding molar fractions

Points d) through h) are repeated for the different T and P values. Point d) is only repeated when there is no sensitivity analysis, for programs COMP1 and COMP3. At the end, the execution time is printed.

CHAPTER VI: RESULTS AND DISCUSSION

The four computer programs described in Chapter V were tried on three problems : 5, 6 ,6B and 7 of Appendix C. We could thus compare the perfomances of dual vs. primal formulations, sensitivity analysis vs re-optimization. The results are presented and discussed in the present Chapter.

This Chapter is organized in four sections. In section one we try to evaluate the perfomance of the starting point routine by comparing the points generated by the routine to the optimum values. In the second section we compare the primal and dual problems. In section three we compare sensitivity analysis and re-optimization. In section four we deal with the effects of incorporating new species to a chemical equilibrium model.

Evaluation of the starting point routine

If the user does not have a good first guess, a starting point is calculated within our programs. The calculation of a dual starting point is based on approximating the dual problem to a linear program. The method described by Lidor (1975) is used, and it is implemented in subroutine LIPSU 2. An account of the method, and of the subroutine LIPSU 2 was made in Chapters IV and V. When a primal starting point is needed, the dual starting point is determined as above, and then it is transformed to a primal point. Subroutine SINGV, described in Chapter IV, performs this task.

Table VI-1 shows the values of the starting points calculated by the programs for problems 5 and 6, and their optimum values.

We will now discuss the results of Table VI-1.

a) The values of the objective functions at the starting points are within 2-4% of the optimal values, which looks like a quite good approximation.

b) The dual variables of the species that are present in greater amounts at equilibrium are determined within 10-20% of their optimal value. The rest of the variables are wide apart. The starting point routine only gives an estimate of as many species as the rank of the exponent matrix A, which should be M in a well posed model. These species correspond to the basic variables of the linear program, and most of the time the dummy species introduced in the linear program to account for non-linearities is one of the basic variables. The rest of the species are given a fixed positive number, regardless of their relative importance. So, if the problem we are dealing with has one or two species present as trace quantities, the starting point does not allow us to distinguish them from other secondary species. See problem 6 in table VI -1. As a consequence, the goodness of the starting point will depend strongly on the specific problem.

c) The primal variables fall within 1.5% to 50% of their optimal values. Again, the range is a characteristic of the particular problem.

d) The primal starting points calculated with this method are

Table VI-1. Evaluation of the starting point routines. Composition, objective function and primal constraint values at the starting point and at optimum for problems 5* and 6**.

Prob.	N. of moles		Z		G/RT	P. const.
	S.P.	Opt.	S.P.	Opt.	S.P./ Opt.	S.P./ Opt.
5*	3.	.0204	-10.0606	-11.4153	746.3633	230.6
	3.	.0007				
	3.	.0153	-22.3678	-20.1267		
	5.5	3.973			777.6382	1.00001
	3.	.0819	-13.5499	-15.8052		
	3.	2.918				
	21.5	19.98	-11.5819	-11.6971		
	3.	.0269				
	5.25	.0338				
	3.	.0005				
6**	.2	.0003	-1.4607	-2.7906	148.1885	49210.
	.3	.0006				
	1.	.9989	-66.8466	-66.4363	150.8133	1.0000
	.2	E-9				
	.2	E-9	-18.0296	-18.6591		
	.2	.0005				
	.2	.0371	-11.9268	-11.7611		
	1.98	1.880				

* Dinkel and Lakshmanan, 1977. P= 40 at. T = 2200 K
** Bonsu, 1981. P= 1 at. T=355 K
N. = Number Z = ln-transformed primal variables
G/RT = Objective function S.P. = Starting point
Opt. = optimum P. Const. = primal constraint

infeasible. In fact, the values of the primal constraints at the starting points are quite far apart from their optimal values. GRG takes care of the infeasibility of the primal starting point by optimizing the sum of the constraint violations in a "Phase I" procedure. We found that the combination of our starting point procedure plus GRG 's "Phase

I" produced better results than a random point. The starting point routines performed well in problems 5,6,6B,7 (Appendix B) however, if a better starting guess is available, the user is encouraged to avoid the starting point routine. Better guesses shorten computation time.

Primal-dual comparisons

To compare the primal and dual formulations of the chemical equilibrium problem,we used the codes COMP1 and COMP2 described in the previous chapter. The problems tested were 5,6,6B, and 7 from appendix B. The values of the objective functions and composition obtained are in appendix B. The execution times are in table VI-2.

Table VI-2. Execution times (in sec.) of primal and dual codes for a fixed P and T.

Problem	Species	Elements	Primal (sec.)	Dual (sec.)
5	10	4	.1265	.1612
6	8	4	.1342	.1695
6b	8	4	.1355	.1720
7	24	4	.5886	failed

To evaluate the relative effectiveness of programming codes we need some criteria. Himmelblau (1972) proposed and discussed several evaluation criteria . We will repeat them here, and then we will try to see how these criteria apply to our problem. The criteria are as follows: i) size of the problem, ii) time required to introduce data into the program ,iii) simplicity of the computer program, iv) success in solving real world problems most of the time, v) accuracy of the results, vi) time.

The size of the problem favours the primal over the dual formulation for the chemical equilibrium problem with many reactions. The time required to introduce data into the program is the same for both the primal and dual problems. The dual program is more simple to implement than the primal, because it does not need subroutines SINGV and SKIP . The primal could be simplified if the subroutine GRG 2 would have the Lagrange multipliers for the primal constraints as arguments, thus subroutine SKIP can be avoided.

As for "solving real-life problems most of the time", the primal always did so. The dual failed when trying a problem with 24 variables (problem 7). The problem had many trace variables, which were sent to the lower boundaries after a few iterations and would not move thereafter. Making the values of the boundaries smaller did not improve the situation. A new computer code, recently available at UBC , MINOS, is based on an algorithm similar to GRG for the case of linear equality constraints. Murtagh and Saunders (1978) claim that MINOS solved a badly scaled dual chemical equilibrium problem of 45 variables. We did not try this code ,though.

The starting point and the required accuracy of the solution influence greatly on the computation time. Since the primal starting point is calculated from the dual, we are almost sure that our comparison between the primal and the dual formulations will not be affected by the starting point. This is only a good approximation because the relations between primal and dual variables that we used in subroutine SINGV only hold as equalities at the optimum point. As for the accuracy of the solution, we used the same parameters in GRG for the primal and dual problems. The results were comparable with the literature, and between the two codes, within .2%. For the primal problem the accuracy for the composition values was determined by the tolerance of the constraint (EPNEWT). The primal constraints state that the sum of the molar fraction of the species present at a particular phase should be one at equilibrium. A value of EPNEWT of 10^{-6} implies that the absolute errors of the molar fractions of each species will be of this order of magnitude. That means that the relative errors for the species present in trace concentrations will be quite important. We can change the value of EPNEWT to improve the accuracy; in fact we tried up to a value of 10^{-10} for problem 7. Still, for this particular problem, the concentration of the species differ as much as 10^{22}. We do not think it is possible to obtain such accuracy at the moment; we are already working with double precision. Besides, we are interested on a code that may solve a wide variety of examples, rather than produce the best solution for a particular problem. The increase in accuracy also means increasing the execution time.

We have shown that our primal and dual codes both start from approximately the same point and get the same results. The time invested in that procedure is an important criterion of the effectiveness of a code. Time refers either to the number of function evaluations or to the execution time. In our case, there is no point in comparing the number of function evaluations, since the mathematical formulations for the problems are quite different. We have to compare execution times.

The execution times in table VI-2 are not standarized, since we were concerned only with the comparison between our codes. Care was taken to run a set of primal-dual problems one after the other, in order to avoid time-sharing problems. The primal formulation proved to be between 23 and 39% faster than the dual for the case of middle-size problems (8-10 species).

Sensitivity analysis and re-optimization

To be able to evaluate the performance of the sensitivy analysis, we compared the codes COMP1 and COMP2 (primal with and without sensitivity analysis) and COMP3 and COMP4 (dual with and without sensitivity analysis). We tested the codes on problems 5 and 6. Problem 5 had been used in the literature as an example for sensitivity analysis (Dinkel and Lakshmanan, 1977), but only the accuracy of the method had been discussed. We repeated their composition values (see chapter IV). The free energy data for problem 6 are calculated from the Mc Bride and Gordon coefficients (NASA , 1971) as a function of temperature. That allowed us to vary the temperature at small intervals, and to compare the effect of these variations on the accuracy of the results and on the execution times.

Table VI-3. Execution times for sensitivity analysis (S.A.) and re-optimization (R.O)

Problem	Species Elements	N. of P, T cond.	T (K)	R. O. Time(sec.)	S. A. Time(sec.)	Form.	St.
5	10/4	5	-	.9673	.4357	D	.44
5	10/4	5	-	.7299	.2885	P	.50
6	8/4	9	50	1.4361	.3144	P	.51
6	8/4	9	50	1.8665	.4661	D	.44
6	8/4	9	10	1.2710	.3011	P	.47
6	8/4	9	5	1.2243	.3235	P	.42
6	8/4	9	1	1.0373	.3226	P	.35

T intervals of temperature between two optimizations, at P=1at. for problem 6. in problem 5, T and P varied.
P= primal D= dual Form.= formulation cond.= conditions.
St= (R.O time / SA. time)/ n. of P, T conditions)

We will use the same comparison criteria mentioned before. The size of the problem is smaller for sensitivity analysis, since it solves a system of linear equations .instead of using a nonlinear optimization method. Sensitivity analysis needs either more time to introduce data, or more programming, because of the stoichiometric coefficients for the reactions. Re-optimization is easier to program : it is only question of adding a DO loop to the optimization program.

Let us take a look at the accuracy of the sensitivity analysis. We will refer the composition values obtained by sensitivity analysis to the values calculated by the optimization procedure. The latter compare well with the literature. We will define now % relative error of species j as:

$$\%\epsilon_j = \{(\delta_j^* - \delta_j^1) / \delta_j^*\} \, 100 \qquad 6.1$$

Where,

$\%\epsilon_j$: percentage error in the determination with sensitivity analysis of the number of moles of species j, referred to re-optimization

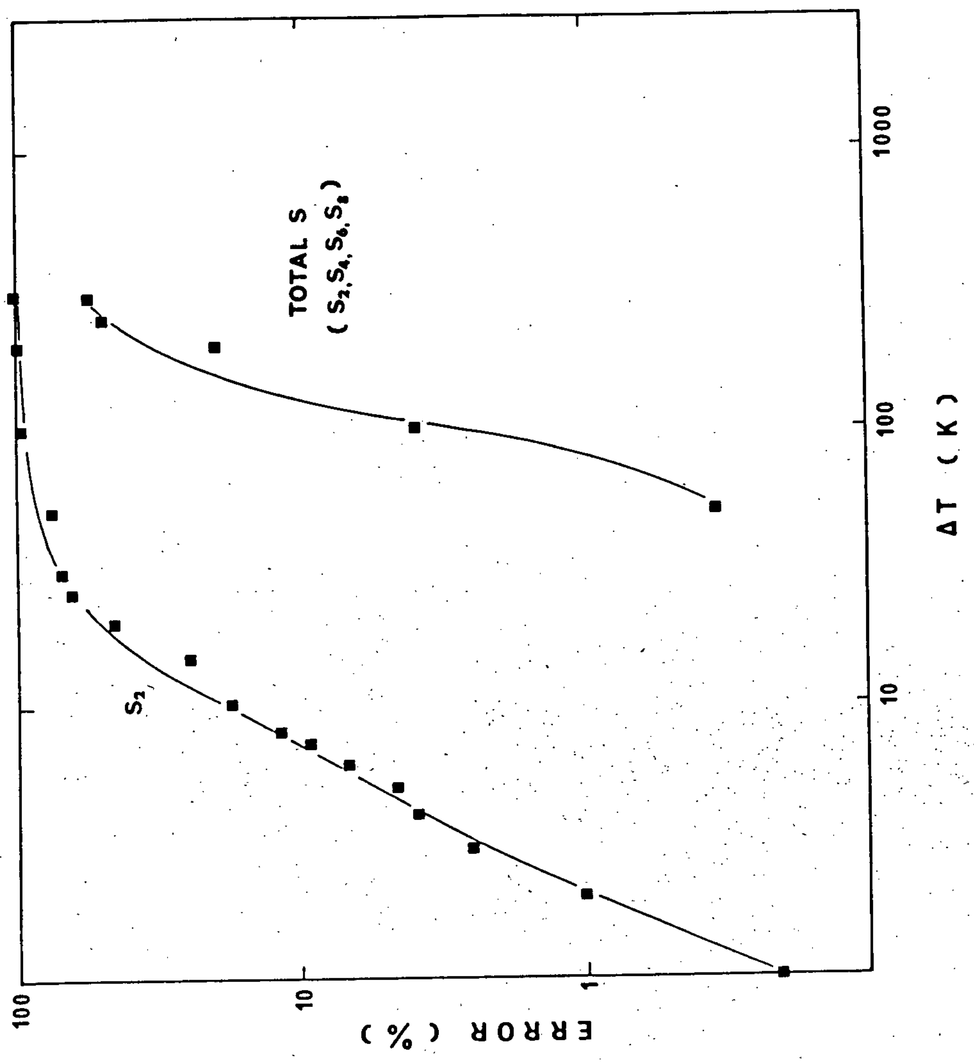
δ_j^1 : number of moles of species j calculated by sensitivity analysis

δ_j^* : number of moles of species j calculated by re-optimization)

The values calculated by re-optimization are, of course, subject to errors. We have discussed in the previous section that the absolute error in the determination of the molar fractions by optimization was around 10^{-6}. That means a considerable relative error for species present as traces.

In figure 1, we plotted % error vs variation of temperature for problem 6. The species number 4 was present in small quantities. Changing the temperature 10K introduced an error of 18% when using sensitivity analysis as compared to a new

Figure 1. % Error vs. Variation of Temperature For Problem 6



optimization. If we calculate the error for a species present in greater quantities, (like the summation of species 4 to 8, which corresponds to total sulphur) the value of the error will be around .2% , similar to the uncertainty of our method. For a variation of 100K, the error for species 4 is around 95%, while the summation of species 4 to 8 only has 4% error.

The values mentioned above for the errors are valid only for this specific problem. The free energy coefficients vary in a different way for different species in different thermodynamical conditions. However, we can assume that the magnitudes of the errors are more influenced by the relative importance of the species than by the relative changes of the free energy coefficients. This is shown in Table VI-4 . We can see there that for a similar variation in the free energy coefficients (water and SO2) the species with lower concentration (SO2) has 33% error when determined with sensitivity analysis, vs. .2% error for the water. Dividing the temperature variation in smaller intervals doesn't seem to have great effect when the total variation of the free energy coefficients is below 10%. It does reduce the error of S2, which has a total variation of the Cj of 41.6%. Thus, as far as accuracy is concerned, we conclude that the sensitivity analysis method should only be used when we are not concerned with the accuracy of the species present in small quantitites, and when the variation of the free energy coefficients are of the order of 10% or less.

Table VI-3 shows the execution times for solving problems 5 and 6 with the four programs COMP1, COMP2, COMP3 and COMP4

Table VI-4. Sensitivity analysis. Effect of the number of temperature increments on the composition, for a total variation of temperature of 50 K. Problem 6*

Species	Xj	%Cj**	% Error (referred to optimization)		
			1 inc.	5 inc.	10 inc.
N_2	.64411	.54	.0001	.00001	.00001
H_2O	.34154	8.1	.23	.24	.12
SO_2	.00049	7.9	33.0	30.4	30.2
S_2	2.e-8	41.6	85.2	74.1	73.0

* Bonsu, 1981
%Cj** % variation of the free energy coefficients for each species, from 410 K (basis) to 460 K.
inc.=increments of temperature. Xj= molar fraction at equilibrium

described in the previous section. To try to account for the different number of P and T conditions considered in the two problems, we defined a ratio St as follows:

$$St = (\text{time for R.O.} \,/\, \text{time for S.A})/\ \text{n.of P,T conditions}$$

where R.O. stands for re-optimization, and S.A. for sensitivity analysis.

The St ratio gives a rough idea of the relative speed of the sensitivity analysis method . We can thus say, very approximately, that the sensitivity analysis method is between 35% to 50% faster than the optimization per number of different thermodynamic conditions.

For problem 6 we see that the times for the primal reoptimizations are shorter for smaller variations of temperature. This can be explained by the fact that, for any new optimization, GRG uses the previous optimal values as starting points. When the temperature differences are small, the optimal values vary very little .

So far, the decision of incorporating a subroutine to perform sensitivity analysis to the final program, will depend on a trade between loss of accuracy and gain of speed. The loss of accuracy seems quite big, and it is not easily predicted. The gain of speed is not very impressive.

Finally, one comment. So far we used the sensitivity analysis routine with stoichiometric coefficients provided by the user. To increase the simplicity of the program to the user, the code should also have a subroutine to calculate these stoichiometric coefficients, using the first optimization results to chose the key components. But this calculation will need some time; and the differences in CPU times between sensitivity analysis and re-optimization will be smaller.

Effects of incorporating new species to a chemical equilibrium model

Problems 6B and 7 are closely related. Problem 6B describes the oxidation of H2S in a Claus reactor with 100% stoichiometric air. Eight species are assumed to be present at equilibrium. Problem 7 is the same case, with 24 species present. The results of solving the primal of both problems are shown in Appendix B for one value of temperature and Pressure. The results agree with the literature in each case. Also, the theoretical composition was checked against experimental values in the literature. We observe the same discrepancies, since our problems agree with their theoretical values.

Both problems, 7 and 6B, have the same quantity of primal

Table VI-5. Effects of adding new species to a model in the objective function and in the total number of moles, for different temperatures.

Temperature	Problem 6b		Problem 7	
(K)	G/RT	λ	G/RT	λ
600	247.24402	6.09281	247.17428	6.08926
650	241.51526	6.13869	241.46424	6.13431
700	236.86576	6.20562	236.83423	6.20244
750	233.11049	6.30549	233.10187	6.31000
800	230.12108	6.44289	230.13785	6.45207

λ : Total number of moles (1 phase, gas)
G/RT: Objective function.

variables. So, for the primal problems, we have the same objective function but the constraint will have tripled the number of terms. For the dual problem, there are three times more variables in problem 7, but we will have the same number of constraints as before. We already mentioned that GRG failed when trying this dual problem.

Table VI-5 shows how the objective function and the total number of moles change with temperature for both problems. We

Table VI-6. Adding new species to a model: effects on the execution time

Problem	Execution time * (sec.) Tolerances= E-6	Execution time* (sec.) Tolerances= E-10
6b	1.6384	-
7	4.2558	5.9839
6	1.4361	-
7 mod**	4.1532-	

* Over 9 different P and T conditions
** Problem 7 was modified to have the same amount of elements as problem 6.

can see that the values are similar; the objective function of problem 7 is smaller than that of problem 6B. Figures 2 and 3 present the variation with T of the composition of the two problems at a range of temperatures. By watching the two figures, we can see that the species present in bigger quantities remain in approximately the same composition in both cases. The main differences arose from the sulfur compounds. It is not only a question of not including species; the free energy coefficients for S4 was different in the two examples. The rest of the species were not important in the range of temperatures considered. The computation times for nine different values of P and T are presented in table VI-6. The execution time for the problem with 24 species is 30% higher than for the problem with 8 species, based on a "per optimization" basis. When solving problem 7 on the same conditions as above, but increasing the accuracy of the primal

Figure 2. Composition vs. Temperature For Problem 6

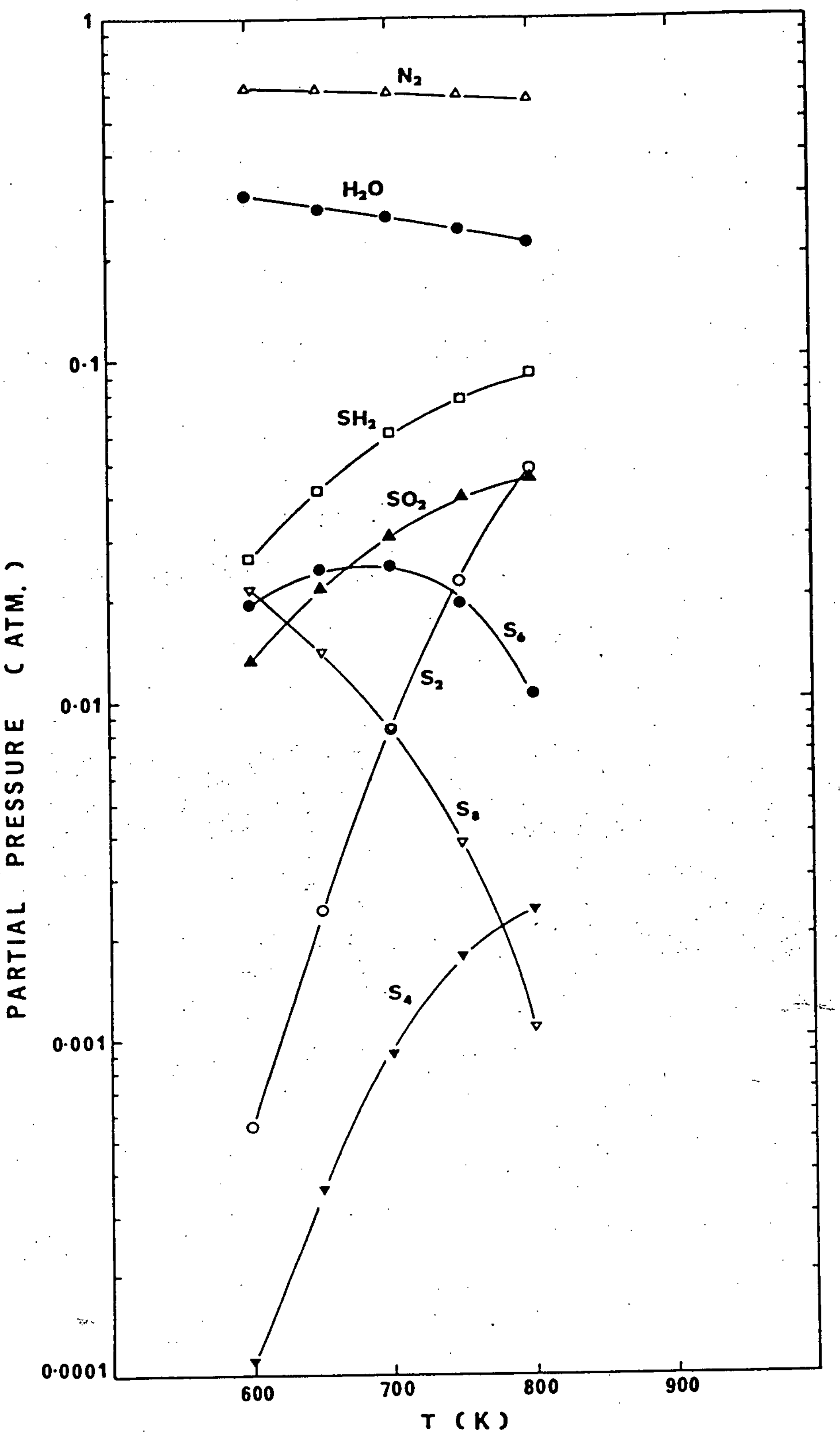


Figure 3. Composition vs. Temperature For Problem 7

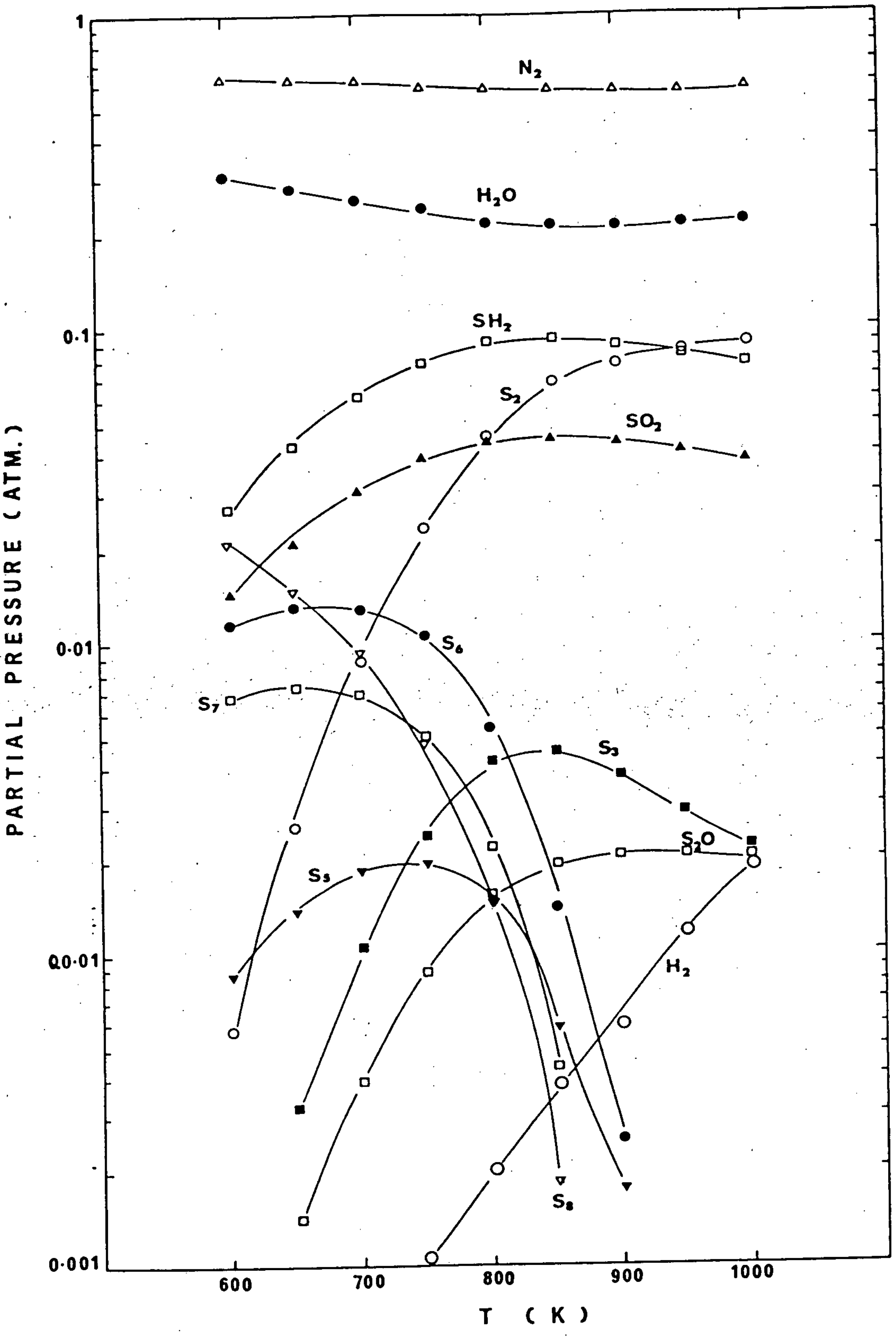


Table VI-7. Adding new species to a model: effects on the composition. Problems 6 and 7, same initial composition, T=600K and P=1 atm.

Species	Molar fractions		Species	Molar fractions	
	Prob. 6	Prob.7		Prob.6	Prob.7
SO2	.0112133	.0116286	H2	-	.0000069
H2S	.0224051	.0232480	H	-	E-19
H2O	.3173050	.3163954	SO	-	E-11
S2	.0003625	.0003916	OH	-	E-20
S4	.0000480	E-22	SO3	-	E-13
S6	.0057975	.0034403	SN	-	E-19
S8	.0040892	.0040944	S2O	-	.0000425
N2	.6387804	.6387978	NO	-	E-22
NH3	-	E-10	S3	-	.0000425
S	-	E-15	S5	-	.0003127
SH	-	E-12	S7	-	.0016156
O	-	0	O2	-	0

prob. 6 : problem 6 of Appendix B.
prob. 7 : problem 7 of Appendix B.

constraint from 10^{-6} to 10^{-10}, the execution time increased to 5.9839, 16% more on a per optimization basis.

Since problem 6 and 6B only differed on the amount of air used (the B vector of amount of elements), we compared problem 6 with problem 7 modified so that the initial conditions of problem 7 were the same as for problem 6. The results are in table VI-7 for T=600K and P= 1at. We observed the same type of variations as with problems 6B and 7 (unmodified): slightly less value of the objective function when more species are added to the model ; no appreciable variation on the composition of the species present in greater amounts.

CHAPTER VII: CONCLUSIONS AND RECOMMENDATIONS

Conclusions

When solving the chemical equilibrium problem with the computer code Generalized Reduced Gradient, we found:

1) GRG cannot solve the primal geometric programming formulation of the problem directly. Either a scaling of the variables and of the objective function ,or a logarithmic transformation has to be performed before using GRG. We found that GRG needs less iterations and function evaluations to solve the logarithmic-transformed problem, than to solve the scaled version. We worked with the transformed primal thereafter.

2) When solving the dual geometric programming with GRG, the objective function needs to be scaled, or the logarithm of the objective function should be minimized. We did not find any particular computational advantage for any of the two approaches, but we chose the latter to compare the results to those of the transformed primal.

3) In both the primal and the dual formulations, the objective function is not very sensitive to changes in the variables. Two values of the objective function may differ in their ninth significant figure, yet the variables differ in their sixth significant figure.

4) The accuracy of the primal solution depends mainly on the tolerances for the constraints. At the optimum, each constraint accounts for the summation of the molar fractions of each species present in the phase described by the constraint. The absolute error of the constraint equals to the summation of the absolute error of each term. Hence any species whose molar fraction is smaller than the tolerance of the constraint will be subject to errors.

5) The accuracy of the dual variables depends on the tolerances for the constraints (conservation of elements) and on the accuracy of the total amount of each element.

6) GRG solves the primal problem 30% faster than the dual,for problems with six reactions. GRG failed to solve the dual of a large problem, with twenty reactions. Adding new species to a model is more easily done with the primal formulation. The only advantage of the dual seems to be that it is more easy to implement than the primal. We concluded that the primal is superior to the dual when both are solved with the GRG code.

7) The method to perform sensitivity analysis suggested by Dinkel and Lakshmanan (1977) proved a thousand times slower than the method derived by Rijckaert (1974). Both methods were tried on a chemical equilibrium problem with six reactions. We used the Rijckaert's method from then on.

8) Sensitivity analysis was between 30% to 50% faster than reoptimization, per number of pressure and temperature changes. However, the results obtained with sensitivity analysis showed discrepancies when compared to the optimization ones. The errors were more serious for the less important species. It is our opinion that the computational speed gained with sensitivity analysis does not compensate for the loss of accuracy of the results.

Recommendations

Based on the conclusions stated above, it is recommended the use of the code COMP1, listed on Appendix C. It is a primal based code, and it does not perform sensitivity analysis.

It is also recommended that further research should be carried on with a better implementation of GRG for handling large dual problems. Murtagh and Saunders (1978) claim that their code MINOS can do so.

REFERENCES CITED

Aris, R. 1970. American Scientist, 58:419-427.

Ballard, D.H. Jelinek, C.O. and Schinzinger, R. 1974 . Computer J. 17:261-266. n.3

Beck, P.A. and Ecker, J.G. 1975 . J.O.T.A. 15:189-201. n.2

Beightler, C.S. and Phillips, D.T. 1976. " Applied Geometric Programming " , John Wiley and Sons, N.Y.

Bennett, H.A. 1979. Ph.D. Thesis, University of British Columbia, Dept. of Chem. Eng. Vancouver, B.C.

Bigelow, J.H. and Shapiro, N.Z. 1971. RAND Corporation Paper P-4628

Bird, C. 1979. "UBC-MATRIX. A Guide To Solving Matrix Problems", UBC Computing Centre, Vancouver, B.C.

Blau, G.E. and Wilde, D.J. 1971 . AIChE Journal, 17:235-240. n.1

Bonsu, A.K. 1981. Ph.D. Thesis, University of British Columbia, Dept. of Chem. Eng. Vancouver, B.C.

Bradley, J. 1973. "An Algorithm For The Numerical Solution of Prototype Geometric Programming", Report of the Institute Of International Research And Standards, Dublin, Ireland.

Brinkley, S.R. 1947. J. Chem. Phys. 15:107-110.

Buzby, B.R. 1974. "Techniques And Experience Solving Really Big Non-linear Programs" in "Optimization Methods", Cottle, R.W. and Krarup, J. Ed. English Univ. Press, London.

Cavallotti, P. Celeri, G. and Leonardis, B. 1980. Chem. Eng. Sci. 35:2297-2304.

Clasen, R.J, 1965. RAND Corporation Memo RM-4345-PR (January, 1965)

Dembo, R.S. 1976 . Math. Prog. 10:192-213.

Dembo, R.S. 1978 . J.O.T.A. , 26:149-183. n.2

Dembo, R.S. 1978 . J.O.T.A. , 26:243-252. n.2

Dembo, R.S. 1979 . Math. Prog. 13:156-175.

Denbigh, K. 1966. "The Principles Of Chemical Equilibrium", Cambridge University Press, 493 pp.

Dinkel, J. and Kochenberger, G. 1977. Op. Res. 25:155-163.

Dinkel, J. and Kochenberger, G. 1979 . Math. Prog. 17:109-113.

Dinkel, J. and Lakshmanan, R. 1975. J. of Eng. Math. 9:343-352.

Dinkel, J. and Lakshmanan, R. 1977. Computers and Chemical Eng. 1:41-47.

Dinkel, J. Elliot, W. and Kochenberger, G. 1977 . Math. Prog. 13:200-220.

Dinkel, J. Kochenberger, G. and Mc.Carl, B. 1974 . Math. Prog. 7:181.

Duffin, R. Peterson, E. and Zener, C. 1967. " Geometric Programming. Theory And Application." , John Wiley and Sons, N.Y. , 278 pp.

Duffin, R.J. And Zener, C. 1969. Proc. of Nat. Ac. of Sci. 63: 629-636.

Duffin, R.J. and Zener, C. 1970. J. Phys. Chem. 74:2419-2423. n.12

Ecker, J.G. 1972 . J.O.T.A. 9:176-181. n.3

Garcia, A. and Hogg, G. 1973 . "Computer Code For The Blau Algorithm" in "Optimizing With FORTRAN ", McGraw Hill, N.Y. pp 136-154.

Gautam, S. and Seider, W.D. 1979. AIChE Journal 25:991-998. n.6.

Gochet, W. and Smeers, Y. 1974 . Cahiers Du Centre D'Etudes De Recherche Operationnelle, 16:23-36.

Gochet, W. Loute, E. Solow, D. 1974 . Cahiers Du Centre D'Etudes De Recherche Operationnelle, 16:469-486.

Gordon, S. and McBride, B.J. 1971. NASA SP- 273, Washington, D.C.

Griffith, R.E. and Stewart, R. A.1961. Management Science, 7: 379-381.

Haarhoff, P.C. and Buys, J.D. 1970. Computer Journal, 13:178-184.

Himmelblau, D.M. 1972. "Applied Non Linear Programming", McGraw-Hill ed.

Kochenberger, G. Woolsey, R. and McCarl, B. 1973 . Op. Res. Q. 24:285-293. n.2

Lasdon, L.S. 1981. "A Survey Of Nonlinear Programming Algorithms And Software", in "Foundations Of Computer Aided Chemical Process Design", Mah, R. S.H. and Seider, W. D. Editors, Pub. by the Nat. Sci. Foundation, NY,NY. pp.185-217.

Lidor, G. and Wilde, D.J. 1978. J.O.T.A. 26:77-96. n.1

Lidor, G. 1975. Ph.D Thesis, Stanford University, Dept. Of Operations Research, Tech. Report 75-8.

McBride, B.J. Heimel, S. Ehlers, J.G. and Gordon, S. 1963. NASA SP-3001, Washington, D.C.

McGreggor, D.E. 1971. Ph.D. Thesis, University Of Alberta, Dept. Of Chemical And Petroleum Eng. Edmonton, Alberta.

McNamara, J.R. 1976 . Op. Res. 24:15-25. n.1

Murtagh, B.A. and Saunders, M. A. 1978. Math. Programming, 14: 41-72.

Passy, U. and Wilde, D. 1968. SIAM J. Appl. Math. 16:363-373. n.2

Passy, U. and Wilde, D. 1969. J. of Eng. Math. 3:325. n. 4

Passy, U. 1972. J.O.T.A. 9:221-234. n.4

Patterson, M. 1981. "UBC-LIPSUB. A Linear Programming Subroutine", UBC Computing Centre, Vancouver, B.C.

Perry, R.H. and Chilton, C.H. 1973. "Chemical Engineering Handbook", McGraw-Hill Chemical Eng. Series, 5th. Ed.

Phillips, D.T. 1974. AIIE Transactions 6:114-119. n.2

Ratner, M. Lasdon, L.S. and Jain, A. 1978. J.O.T.A. 26:253-263. n.2

Richter, E. and Hofmann, H. 1976. Internat. Chem. Eng. 16:137-143.

Rijckaert. M. 1974. "Sensitivity Analysis In Geometric Programming" in "Mathematical Programming For Activity Analysis", Ed. P. Van Moeseke, North Holland Pub.Co. Amsterdam, Holland, pp. 61-72

Rijckaert, M. and Martens, X.M. 1976 . Math. Prog. 11:89-93.

Rijckaert, M.J. and Martens, X.M. 1978 . J.O.T.A., 26:325-340. n.2

Sarma, P.V. Martens, X.M. Reklaitis, G.V. and Rijckaert M.J. 1978 . J.O.T.A. 26:185-242. n.2

Schneider, D.R. and Reklaitis, G.V. 1975. Chem. Eng. Sci. 30: 243-247.

Schubert, E. and Hofmann, H. 1976. Internat. Chem. Eng. 16:132-136. n.1

Scully, D.B. 1962. Chem. Eng. Sci. 17:977-985.

Shapiro, N.Z. 1964. RAND Corporation Report RM-4128-PR

Shapley, M. and Cutler, L. 1970. RAND Corporation Report R-495-PR (June, 1971)

Van Zeggeren, F. and Storey, S.H. 1970. "The Computation Of Chemical Equilibria", Cambridge University Press, 176 pp.

Wales, K. 1977. "UBC-GRG. Generalized Reduced Gradient Program", UBC Computing Centre, Vancouver, B.C.

Waller, K. and Maklla, P. 1981. Ind. Eng. Chem. Process Des. Dev. 20:1-11. n.1

Warga, J. 1963. J. Soc. Indust. Appl. Math. 11:594-606.

Westerberg, A.W. 1981. "Optimization in Computer Aided Design", in "Foundations of Computer Aided Chemical Process Design", Mah, R.S.H. and Seider, W.D. Editors. Pub. by the Nat. Sci. Foundation, NY,NY. Pp.149- 183.

White, W. Johnston, S. and Dantzig, G. 1958. J. Chem. Phys. 28: 751-755.

Zeleznik, F.J. and Gordon, S. 1968. Ind. Eng. Chem. 60:27-57. n.6

NOMENCLATURE

a_j	= Activity of species j.
A	= Augmented exponents matrix.
AA	= Exponents matrix. AAij atom grams of element i are present in one mole of species j.
Bi	= Amount of atom grams of element i.
Cj	= Free energy coefficient of species j. For gases, $Cj = \mu_j^{\circ}(T)/RT + \ln P/1$ atm. ; for condensed phases, $Cj = \mu_j^{\circ}(T,P)/RT$
c_j	= Geometric program (GP) coefficients; $c_j = \exp(-Cj)$
D	= Number of independent chemical reactions.
g_o	= Primal GP objective function.
g_κ	= Primal GP constraints.
G	= Gibbs free energy. Gj = Molar Gibbs free energy.
h	= Objective function of the transformed primal GP. $h = \ln g_o = G/RT$
I	= Identity matrix.
J	= Hessian of ln v.
K	= Number of phases at equilibrium.
Kd	= Equilibrium constant for reaction d.
M	= Number of elements.
N	= Number of chemical species at equilibrium.
P	= Pressure.
r_d	= Extent of reaction d.
R	= Universal gas constant.
Sj	= Vector of Gordon and McBride coefficients for species j
t_i	= Primal variable asociated with element i
T	= Temperature.
Ujd	= Stoichiometric coefficient of species j for

reaction d.

v	= Dual GP objective function. $-\ln v = G/RT$
Xj	= Molar fraction of species j at equilibrium.
y	= Variable of the linear program approximation to the dual GP.
Zi	= Ln-transformed primal variable. $Z = \ln t$

GREEK LETTERS:

α = Parameter in unidimensional search for GRG.

δ_j = Number of moles of species j.

$\delta_j{}^o$ = Initial number of moles of species j.

$\%\epsilon$ = Percentual error of the composition determined by sensitivity analysis, referred to re-optimization.

λ_K= Total number of moles in phase k.

μ_j= Chemical potential of species j in phase k.

$\mu_j{}^o$ = Reference chemical potential.

Π_K= Lagrange multiplier for the k-primal constraint. When solving the transformed primal problem, $\Pi_K = \lambda_K$

ξ = Variable of the linear program approximation to the dual GP. It accounts for non-linearities.

APPENDIX A: COMPUTER PROGRAMS

PROGRAM JOTA

```
C
C
C
C THIS PROGRAM PERFORMS SENSITIVITY ANALYSIS BY INVERTING THE
C JACOBIAN MATRIX
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DA(20,20),DT(20,20),IPERM(40)
      DIMENSION U(20,20),DELT(20),C(20),C1(20),SUM(20,20)
      DIMENSION SUMA(20),SUMB(20),SUMC(20)
C READ THE DATA
      READ (5,10) N,NDIMA,NDIMT
10    FORMAT (3I3)
      READ (5,20) N1,N2,OPTIM
20    FORMAT (2I3,F10.0)
      READ (5,30) (DELT(I),I=1,N2)
30    FORMAT (6F10.0)
      READ (5,40) (C(I),I=1,N2)
40    FORMAT (6F10.0)
      READ (5,50) (C1(I),I=1,N2)
50    FORMAT (6F10.0)
      READ (5,60) ((U(I,J),J=1,N2),I=1,N)
60    FORMAT (12F4.0)
C   FORM THE J MATRIX
      DO 9 I=1,N
      DO 9 J=1,N
      SUM(I,J)=0.D0
      DO 8 K=1,N1
8     SUM(I,J)=SUM(I,J)+U(I,K)*U(J,K)/DELT(K)
      WRITE (6,200)  I,J,SUM(I,J)
200   FORMAT (' I=',I4,' J=',I4,' SUM(I,J)=',G20.12)
      DA(I,J)=SUM(I,J)-U(I,N2)*U(J,N2)/DELT(N2)
9     CONTINUE
C WRITE THE J MATRIX
      WRITE (6,70)
70    FORMAT (' MATRIX J')
      WRITE (6,80)((DA(I,J),J=1,N),I=1,N)
80    FORMAT (1X,6G14.6)
C CALCULATE THE TIME
      TIME=SCLOCK(0.0)
C CALCULATE THE INVERSE
      CALL INV(N,NDIMA,DA,IPERM,NDIMT,DT,DDET,JEXP,DCOND)
C WRITE THE EXECUTION TIME
      WRITE (6,900) TIME
900   FORMAT (' EXECUTION TIME IS',F10.5)
      IF (DDET) 1,2,1
C WRITE THE INVERSE
1     WRITE (6,90) DCOND
90    FORMAT (' COND NO.=',G20.12,' INVERSE')
      WRITE (6,100)((DT(I,J),J=1,N),I=1,N)
100   FORMAT (1X,6G14.6)
C CALCULATE THE NEW OPTIMUM
```

```
      SUMD=0.D0
      DO 3 I=1,N1
3     SUMD=SUMD+DELT(I)*(C1(I)-C(I))
      OPTIM=OPTIM+SUMD
C WRITE THE NEW OPTIMUM
      WRITE (6,120) OPTIM
120   FORMAT (' NEW OPTIM.=',6G14.6)
C CALCULATE THE NEW SOLUTION
      DO 12 I=1,N2
      SUMC(I)=0.D0
      DO 11 L=1,N
      SUMA(L)=0.D0
      DO 21 K=1,N
      SUMB(K)=0.D0
      DO 22 J=1,N1
22    SUMB(K)=SUMB(K)+U(K,J)*(C1(J)-C(J))
21    SUMA(L)=SUMA(L)+DT(L,K)*SUMB(K)
11    SUMC(I)=SUMC(I)+SUMA(L)*U(L,I)
12    DELT(I)=DELT(I)+SUMC(I)
C WRITE THE NEW SOLUTION
      WRITE (6,130)
130   FORMAT (' NEW SOLUTION')
      WRITE (6,140) (DELT(I),I=1,N2)
140   FORMAT (1X,6G14.6)
      STOP
2     WRITE (6,110)
110   FORMAT (' INVERSION FAILED')
      STOP
      END
```

Program NPLUSK

```
C
C
C
C THIS PROGRAM PERFORMS SENSITIVITY ANALYSIS BY SOLVING A
C SYSTEM OF LINEAR EQUATIONS
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION DA(12,12), DT(12,12), DB(12), DX(12),
     1        DRZ(12), IPERM(24), A(12,12), DELT(12)
      DIMENSION C(12), C1(12), U(12,12), SUM(12),V(12)
      READ (5,10)  N1,K,M
10    FORMAT (3I3)
      READ (5,20) N,NDIMAT,N2
20    FORMAT (3I3)
      DEPS=1.D-14
C READ IN DATA TO FORM THE (N1+KXN1+K) MATRIX
      READ (5,30) ((A(I,J),J=1,N),I=1,M)
30    FORMAT (12F4.0)
      READ (5,40) (DELT(I),I=1,N)
40    FORMAT (6F10.0)
      READ (5,50)  (C(I),I=1,N)
50    FORMAT (6F10.0)
      READ (5,60) (C1(I),I=1,N)
60    FORMAT  (6F10.0)
      READ (5,70) ((U(I,J),J=1,N),I=1,N2)
70    FORMAT  (12F4.0)
      NRHS=1
      ITMAX=14
C FORM THE FINAL MATRIX
C FIRST THE NORMALITY CONDITION
      DA(1,1)=1.D0
      DO 1 J=2,N
1     DA(1,J)=0.D0
C NOW THE ORTHOGONALITY COND.
      DO 2 I=1,M
      DO 2 J=1,N
      L= I+1
2     DA(L,J)=A(I,J)
C NOW,SUMMATION OF NUMBER OF MOLES
      DO 4 I=1,2
      L=I+M+1
      DA(L,1)=0.D0
      DO 5 J=2,N1
5     DA(L,J)=1.D0
6     DA(L,N)=-1.D0
4     CONTINUE
C NOW, THE EQUILIBRIUM CONDITIONS
      DO 7 I=1,N2
      DO 7 J=1,N
      L=I+M+1+K
7     DA(L,J)=U(I,J)/DELT(J)
C NOW CALCULATE THE RIGHT HAND SIDE VECTOR
```

```
      M3=1+M+K
      DO 8 I=1,M3
8     DB(I)=0.D0
      DO 11 I=1,N2
      SUM(I)=0.D0
      DO 9 J=1,N1
9     SUM(I)=SUM(I)+U(I,J)*(C1(J)-C(J))
      L=I+M+1+K
11    DB(L)=SUM(I)
C WRITE THE DA MATRIX AND THE DB VECTOR
      WRITE (6,850)
850   FORMAT (' MATRIX OF COEFFICIENTS')
      WRITE (6,90) ((DA(I,J),J=1,N),I=1,N)
90    FORMAT (1X,6G12.6)
      WRITE (6,950)
950   FORMAT (' B VECTOR')
      WRITE (6,100) (DB(I),I=1,N)
100   FORMAT (1X,6G12.6)
C CALCULATE THE TIME OF EXECUTION
      TIME=SCLOCK(0.0)
C SOLVE THE SYSTEM
      CALL DSLIMP(DA,DT,DB,DX,DRZ,IPERM,N,NDIMAT,DEPS,NRHS,ITMAX)
      TIME=SCLOCK(TIME)
C WRITE THE EXECUTION TIME
      WRITE(6,800) TIME
800   FORMAT (' EXECUTION TIME IS ',F10.5)
C WRITE OUT RESULTS
      WRITE (6,900)
900   FORMAT (' VARIATION OF NUMBER OF MOLES')
      WRITE (6,110)  (DX(I),I=1,N)
110   FORMAT (1X,6G14.6)
C CALCULATE THE NEW NUMBER OF MOLES
      DO 15 I=1,N
15    V(I)=DX(I)+DELT(I)
      WRITE (6,120)
120   FORMAT (' SOLUTION')
      WRITE (6,130) (V(I),I=1,N)
130   FORMAT (1X,6G14.6)
      STOP
      END
```

Program COMP1

```
C
C
C THIS PROGRAM SOLVES THE PRIMAL GP FOR THE CHEMICAL EQUILIBRIUM
C PROBLEM. IT CALLS SUBROUTINE LIPSU2 TO GET A FIRST DUAL POINT
C FOR THE OPTIMIZATION. THEN IT CALLS SUBROUTINE SINGV WHICH
C CALCULATES THE  FIRST PRIMAL STARTING POINT. SUBROUTINES GRGIN,
C GRG2 AND GRGEG, FROM  UBC,SOLVE THE OPTIMIZATION PROBLEM.
C SUBROUTINE GCOMP CALCULATES THE OBJECTIVE FUNCTION AND THE
C CONSTRAINTS, AND IS CALLED FROM GRG. THE PROCEDURE IS
C REPEATED FOR DIFFERENT TEMPERATURES AND PRESSURES.
C SUBROUTINE FREEN CALCULATES THE FREE ENERGY COEFFICIENTS
C (G/RT), USING THE DATA FROM GORDON AND MC. BRIDE
C
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL*1 BLANK /' '/, SEMIC /';'/
C DIMENSION STATMENTS . LABELLED COMMONS ARE USED.
C N=NUMBER OF CHEMICAL SPECIES   +1
C M=NUMBER OF CHEMICAL ELEMENTS   (ATOMS) + 1
C K=NUMBER OF PHASES
C NN= NUMBER OF DIFFERENT TEMP. AND PRESSURE CONDITIONS.
C NE=1 IF THE FREE ENERGY COEFFICIENTS ARE ALREADY AVAILABLE.
C IF THEY ARE TO BE CALCULATED BY THE GORDON AND MC.BRIDE
C COEFFICIENTS, USE ANY OTHER INTEGER.
C NF=1 IF THE DUAL STARTING POINT ROUTINE IS TO BE USED.
C OTHERWISE, EQUAL NF TO ANY OTHER INTEGER, AND PROVIDE
C A DUAL STARTING POINT. DON'T FORGETTHAT
C THE FIRST DUAL VARIABLE IS DUMMY AND EQUAL TO 1.D0.
C A(I,J),I=1,M,J=1,N = EXPONENTS MATRIX. A(1,J) CORRESPONDS
C TO THE NORMALITY CONDITION
C C(JJ,L),JJ=1,N, L=1,NN FREE ENERGY COEFFICIENTS OF SPECIES JJ
C T,P=EQUILIBRIUM TEMPERATURE AND PRESSURE
C X(I)=Z(I,1),I=1,M-1 =TRANSFORMED PRIMAL VARIABLES.FOR
C SUBROUTINE LIPSU2,X(J),J=1,N+1, CORRESPOND TO THE VARIABLES
C LINEAR PROGRAM.
C XNEW(J),J=2,N =NUMBER OF MOLES OF SPECIES J; XNEW(1)=1.D0
C IS A DUMMY VARIABLE TO ACCOUNT FOR THE NORMALITY CONDITION.
C XMF(J)=MOLAR FRACTION OF SPECIES J, J=2, 3, N.
C XLAG(I),I=1,K = LAGRANGE MULTIPLIER OF CONSTRAINT I
C S(J,L),J=1,N-1,L=1,7 =COEFFICIENTS FOR GORDON & MC.BRIDE
C POLYNOMIALS TO CALCULATE THE FREE ENERGY COEFFICIENTS.
C B(I),I=2,M = VECTOR OF AMOUNTS OF ELEMENT I.B(1)=1, DUMMY.
C F = OBJECTIVE FUNCTION.
      COMMON/AX/A(10,30)
      COMMON/BX/B(10)
      COMMON/CX/C(30,10)
      COMMON/DX/N,M/HX/K,NN/GX/F/XL/L
      COMMON/EX/T(30),P(30)
      COMMON/FX/X(31)/ZX/Z(10,5)
      DIMENSION XMF(30),VL(30),VH(30)
      COMMON/XNE/XNEW(31)/SX/S(30,10)
      DIMENSION XLAG(10),SUMC(30)
```

```
C
C READ IN DATA
C
      READ(5,10)  N,M,K,NN,NE,NF
10    FORMAT (6I5)
C
C CALCULATE FREE ENERGY COEFFICIENTS BY GORDON AND MC.BRIDE
C
      IF (NE.EQ.1) GO TO 13
      SX =550.D0
C
C CALCULATE THE TEMPERATURE AT FIXED INTERVALS.
C
      DO 20 L=1,NN
      SX=SX +5.D1
      T(L)=SX
20    P(L)=1.D0
      N1=N-1
      READ (5,40) ((S(I,L),L=1,7),I=1,N1)
40    FORMAT (5G13.7)
      CALL FREEN
      GO TO 400
13    READ (5,311) (T(L),L=1,NN)
C
C IF THE C(I,L) MATRIX IS AVAILABLE BY OTHER SOURCES THAN
C GORDON & MC.BRIDE, READ NOW TEMPERATURE, PRESSURE, AND THE
C C(I,L) MATRIX.
C
      READ (5,311) (P(L),L=1,NN)
      READ (5,311) ((C(I,L),L=1,NN),I=1,N)
311   FORMAT (5F10.0)
C
C READ THE EXPONENT MATRIX AND THE B VECTOR.
C
400   READ (5,50) ((A(I,J),J=1,N),I=1,M)
50    FORMAT (9F4.0)
      READ (5,60) (B(I),I=1,M)
60    FORMAT (5F10.0)
C
C PUT NF=1 IF YOU WANT THE PROGRAM TO CALCULATE A DUAL STARTING
C POINT. OTHERWISE, IT WILL NOW READ YOUR FIRST DUAL GUESS.
C
      IF (NF.EQ.1) GO TO 500
      READ (5,511) (XNEW(J),J=1,N)
511   FORMAT (5F10.0)
      TIME=SCLOCK(0.)
      GO TO 600
500   TIME=SCLOCK(0.)
C
C CALLS SUBROUTINE LIPSUB2.IT PROVIDES A STARTING POINT FOR THE
C DUAL PROBLEM.
      CALL LIPSU2
      XNEW(1)=1.D0
      DO 100 J=2,N
```

```
100   XNEW(J) = X(J - 1) + X(N)
      WRITE (6,12) (XNEW(J),J=1,N)
12    FORMAT (' FIRST DUAL POINT'/1X,6G16.8)
C
C SUBROUTINE SINGV WILL NOW TRANSFORM THE DUAL VARIABLES INTO
C PRIMAL ONES.
C
600   CALL SINGV
      M1=M-1
      WRITE (6,16) (Z(I,1),I=1,M1)
16    FORMAT (' FIRST PRIMAL STARTING POINT',4G16.8)
      DO 11 I=1,M1
11    X(I)=Z(I,1)
      DO 202 L=1,NN
      CALL FTNCMD('ASSIGN 5=-DATA;')
      CALL FTNCMD ('ASSIGN 7=*SINK*;')
C OPTIMIZATION IS PERFORMED FOR NN DIFFERENT P AND T CONDITIONS
C A SCRATCH FILE IS CREATED TO WRITE DOWN THE DATA NEEDED FOR
C THE UBC SUBROUTINES GRGIN AND GRG TO PERFORM THE OPTIMIZATION
C
C FIRST ARE THE CONTROL CARDS
C
      IG=0.D0
      WRITE (5,15) M1,K,IG
15    FORMAT (3I6)
C
C NOW WRITE THE LOWER BOUNDS OF THE VARIABLES.WE ADD -30 TO THE
C PRIMAL STARTING POINT.
C
      WRITE (5,25)
25    FORMAT ('LBV=')
      DO 115 I=1,M1
115   VL(I)=X(I)-3.D1
      WRITE (5,35) (BLANK,I,VL(I),I=1,M1),SEMIC
35    FORMAT (6(A1,I3,G10.2))
C NOW WRITE THE UPPER BOUNDS OF THE VARIABLES.WE SET THEM AS 0.0
C
      WRITE (5,55)
55    FORMAT ('UBV=')
      DO 125 I=1,M1
125   VH(I)=0.D0
      WRITE (5,35) (BLANK,I,VH(I),I=1,M1),SEMIC
C
C PRIMAL CONSTRAINTS. IF THERE ARE MORE THAN ONE, THE FORMAT
C SHOULD BE :FORMAT('UBC= 1 1.D0 2 1.D0 K 1.D0 ;')
C
      WRITE (5,66)
66    FORMAT('UBC= 1 1.D0 ;')
      WRITE (5,65)
65    FORMAT ('QUAD')
C
C EPNEWT IS THE TOLERANCE FOR THE PRIMAL CONSTRAINTS.THE ACCURACY
C OF THE RESULTS IS VERY SENSITIVE TO THIS VALUE.
C
```

```
      WRITE (5,75)
75    FORMAT ('EPNEWT=1.D-10')
      WRITE (5,85)
C
C EPSTOP IS THE TOLERANCE FOR THE OBJECTIVE FUNCTION
C
85    FORMAT ('EPSTOP=1.D-6')
      WRITE (5,95)
95    FORMAT ('EPSBOUND=1.D-6')
      WRITE (5,105)
105   FORMAT ('EPSPIV=1.D-6')
      WRITE (5,61)
C
C PRINTCTL MAY BE SET AT HIGHER VALUES (UP TO 4) FOR MORE
C INFORMATION ON THE OPTIMIZATION PROGRESS.
C
61    FORMAT (' PRINTCTL=1')
      WRITE (5,107)
107   FORMAT('X=')
      WRITE (5,109) (X(I),I=1,M1)
109   FORMAT (6G18.6)
      WRITE (5,108)
108   FORMAT ('OPTIMIZE'/'GO'/'STOP')
      REWIND 5
C   NOW GRGIN CAN READ INPUT FROM  -DATA;GRG2 AND GRGRES WILL
C THEIR OUTPUT INTO THE SCRATCH FILE -GRGOUT
      CALL FTNCMD('ASSIGN 6=-GRGOUT;')
      CALL GRGIN(&1,&2)
      CALL GRG2(X,F,&1)
      CALL GRGRES
      DO 22 I=1,M1
22    Z(I,L)=X(I)
      GO TO 2
1     WRITE (7,3)
3     FORMAT (' GRG HAS FAILED')
      GO TO 202
2     REWIND 6
C
C SUBROUTINE SKIP READS THE LAGRANGE MULTIPLIERS FOR EACH PRIMAL
C CONSTRAINT. THEY ARE THE NEGATIVE OF THE TOTAL NUMBER OF MOLES
C IN EACH CONSTRAINT.
C
      CALL SKIP(&202)
      READ (6,26) (XLAG(I),I=1,K)
26    FORMAT (1X,10F13.0)
      WRITE (7,205) T(L),P(L),F
205   FORMAT ('T(K)=',F10.0,'P(AT)=',F10.0,'OBJ.F=',G16.8)
      WRITE (7,215)
215   FORMAT (' MOLAR FRACTIONS')
      DO 33 J=2,N
      SUMC(J)=0.D0
      DO 31 I=1,M1
      II=I+1
31    SUMC(J)=SUMC(J)+A(II,J)*X(I)
```

```
      IF (SUMC(J).GE.0.D0) GO TO 34
      R=DEXP(SUMC(J)-C(J,L))
      GO TO 33
34    R=0.D0
33    XMF(J)=R
      WRITE (7,235) (XMF(J),J=2,N)
235   FORMAT (1X,6G16.8)
      WRITE (7,238) (XLAG(I),I=1,K)
238   FORMAT (' TOTAL NUMBER OF MOLES',G16.8)
      WRITE (7,239)
239   FORMAT ('NUMBER OF MOLES')
      DO 237 J=2,N
237   XNEW(J)=-XLAG(1)*XMF(J)
      WRITE (7,240) (XNEW(J),J=2,N)
240   FORMAT (1X,6G16.8)
202   CONTINUE
23    TIME=SCLOCK(TIME)
      WRITE (7,19) TIME
19    FORMAT (' EXECUTION TIME =',F6.4)
      STOP
      END
C
C SUBROUTINE LIPSU2 CALCULATES A DUAL STARTING POINT
C BY APPROXIMATING THE DUAL PROBLEM TO A LINEAR PROGRAM.
C THE LINEAR PROGRAM ISTHEN SOLVED BY SUBROUTINE
C LIPSUB FROM UBC.
C
      SUBROUTINE LIPSU2
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TABLO(30,30),NVIN(30),NVOUT(30),BBOBJ(20),
     1        UBOBJ(20),BRHS(20),UBRHS(10)
      COMMON/AX/A(10,30)
      COMMON/BX/B(10)
      COMMON/CX/C(30,10)/GX/OPTIM
      COMMON/DX/NVARS,NCONST
      COMMON/FX/X(31)
      N1=NVARS-1
      M1=NCONST-1
      NEQUAL=M1
      MAXIM=0
      IFOBJ=0
      IFRHS=0
      NCOLS=NVARS+1
      NROWS=NCONST+1
C ZERO THE TABLEAU
      DO 200 J=1,NCOLS
      DO 200 I=1,NROWS
200   TABLO(I,J)=0.D0
C FIRST ROW IN THE TABLEAU IS THE OBJECTIVE FUNCTION
      SUM=0.D0
      DO 101 J=1,N1
      JJ=J+1
      TABLO(1,J)=C(JJ,1)
101   SUM=SUM+TABLO(1,J)
```

```
      TABLO(1,NVARS)=SUM
      TABLO(2,NVARS)=-1.D0
      TABLO(2,NCOLS)=-.0001D0
      DO 203 I=2,NCONST
      DO 202 J=1,N1
202   TABLO(I + 1, J) = A(I, J + 1)
      TABLO(I + 1, NVARS) = 1.D0
203   TABLO(I + 1, NCOLS) = B(I)
260   NLVARS = NVARS
      CALL LIPSUB(TABLO, 30, NCONST, NLVARS, NEQUAL, MAXIM,
     1           IFOBJ, IFRHS, TOL, NVIN, NVOUT,BBOBJ,
     2            UBOBJ, BBRHS, UBRHS, &540)
      NP1=NLVARS+1
      OPTIM =TABLO(1,NP1)
      IF (MAXIM.NE.1) OPTIM=-OPTIM
      DO 12 I=1,NVARS
12    X(I)=0.D0
      DO 13 I=2,NROWS
      J = NVIN(I)
13    X(J)=TABLO(I,NP1)
      WRITE (6,320)   (X(J),J=1,NVARS)
320   FORMAT (1X,5G18.6)
      RETURN
540   WRITE (6,560)
560   FORMAT ('0DUAL STARTING POINT ROUTINE FAILED')
      DO 580 I = 1, NVARS
580   X(I) = 10.0D0
      RETURN
      END
C
C SUBROUTINE GCOMP CALCULATES THE VALUES OF THE OBJECTIVE
C FUNCTION AND OF THE CONSTRAINTS
C
      SUBROUTINE GCOMP(G,X)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/AX/A(10,30)/BX/B(10)/CX/C(30,10)/DX/N,M/XL/L
      COMMON/HX/K,NN
      DIMENSION G(1),X(1),SUMF(5),XF(30),XMF(30)
      DO 10 I=1,K
      SUMF(I)=0.D0
      DO 20 J=2,N
      XF(J)=0.D0
      DO 30 II=2,M
      I1=II-1
30    XF(J)=XF(J)+A(II,J)*X(I1)
      IF (XF(J).GE.0.D0)  GO TO 40
      XMF(J)=XF(J)-C(J,L)
      R=DEXP(XMF(J))
      GO TO 20
40    R=0.D0
20    SUMF(I)=SUMF(I)+R
10    G(I)=SUMF(I)
      SUM=0.D0
      DO 50 J=2,M
      JJ=J-1
50    SUM=SUM+B(J)*X(JJ)
      G(K+1) = -SUM
      RETURN
      END
C
C SUBROUTINE SINGV CALCULATES A FIRST STARTING POINT
C FOR THE PRIMAL PROBLEM.IT SOLVES AN OVERDETERMINED
C SYSTEM OF EQUATIONS THAT RELATES PRIMAL AND DUAL VARIABLES.
C THE METHOD USED IS A SINGULAR VALUE DECOMPOSITION.
C
      SUBROUTINE SINGV
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION  A(30,30),V(30,30),S(30)
      COMMON/AX/AD(10,30)/CX/C(30,10)/XNE/XNEW(31)/DX/N1,M1
      COMMON/ZX/X(10,5)/HX/K,NN
      NP=1.D0
      N=N1-1
      M=M1-1
      MNP=M+NP
      NDIMAU=30
      NDIMV=30
      DO 10 I=1,N
      II=I+1
      DO 10 J=1,M
      JJ=J+1
10    A(I,J)=AD(JJ,II)
      SUMX=0.D0
      DO 12 J=2,N1
12    SUMX=SUMX+XNEW(J)
      DO 11 I=1,N
      II=I+1
      DO 11 J=M1,MNP
      IF (XNEW(II).LE.0.D0) XNEW(II)=1.D-16
11    A(I,J)=C(II,1)+DLOG(XNEW(II))-DLOG(SUMX)
      WRITE (6,14) ((A(I,J),J=1,MNP),I=1,N)
14    FORMAT (1X,5G18.6)
      CALL DSLSVD(A,S,V,NDIMAU,NDIMV,N,M,NP,&140)
      EPS=1.D-6
      SS=S(1)*EPS
      DO 60 J=1,M
      IF (S(J).LT.SS) GO TO 70
      DO 50 I=1,M
50    V(I,J)=V(I,J)/S(J)
60    CONTINUE
      J=M+1
      GO TO 90
70    WRITE (6,80) ((V(I,K),I=1,M),K=J,M)
80    FORMAT (1X,4G13.5)
90    IF (J.GT.M) GO TO 120
      DO 110 K=J,M
      DO 110 I=1,M
110   V(I,K)=0.D0
120   MP1=M+1
```

```
      CALL DGMULT(V,A(1,MP1),X,M,M,NP,NDIMV,NDIMAU,10)
      RETURN
140   WRITE (6,150)
150   FORMAT (' ERROR RETURN FROM DSLSVD')
      STOP
      END
C
C SUBROUTINE SKIP WILL READ THE LAGRANGIAN MULTIPLIERS
C FOR THE PRIMAL CONSTRAINTS AS PRINTED BY GRG IN THE SCRATCH
C FILE -GRGOUT.
C
      SUBROUTINE SKIP(*)
      LOGICAL*1 RECORD(150)
      INTEGER*2 LEN
      LOGICAL EQCMP
100   CALL READ(RECORD,LEN,0,LNR,6,&200)
      IF (LEN .LT. 70 .OR. LEN .GT. 75) GO TO 100
      IF (EQCMP (21, RECORD, '0LAGRANGE MULTIPLIERS')) RETURN
      GO TO 100
200   RETURN 1
      END
C
C SUBROUTINE FREEN CALCULATES THE FREE ENERGY PARAMETERS
C FROM THE GORDON AND MC.BRIDE POLYNOMIALS COEFFICIENTS.
C
      SUBROUTINE FREEN
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CX/C(30,10)/SX/S(30,10)
      COMMON/DX/N,M/HX/K,NN
      COMMON/EX/T(30),P(30)
      DO 10 L=1,NN
      C(1,L)=0.D0
      N1=N-1
      DO 10 J=1,N1
      JJ=J+1
10    C(JJ,L)=S(J,1)*(1.D0-DLOG(T(L)))-S(J,2)*T(L)/2.D0-
     1     (S(J,3)*T(L)**2.D0)/6.D0 -(S(J,4)*T(L)**3.D0)/1.2D1
     2    -(S(J,5)*T(L)**4.D0)/2.D1+S(J,6)/T(L)-S(J,7)
     3    +DLOG(P(L))
      WRITE(6,11) ((C(J,L),J=1,N),L=1,NN)
11    FORMAT ('MATRIX OF FREE ENERGY COEFICIENTS',/,1X,4F12.7)
      RETURN
      END
```

Program COMP2

```
C
C
C
C THIS PROGRAM SOLVES THE PRIMAL GP FOR THE CHEMICAL EQUILIBRIUM
C PROBLEM. IT CALLS THE SUBROUTINE LIPSU2 TO GET A FIRST DUAL
C POINT FOR THE OPTIMIZATION. THEN IT CALLS SUBROUTINE SINGV E
C WHICH CALCULATES THE CORRESPONDING PRIMAL
C STARTING POINT. SUBROUTINES GRGIN AND GRG2, FROM UBC,
C SOLVE THE OPTIMIZATION PROBLEM.SUBROUTINE GCOMP CALCULATES THE
C OBJECTIVE FUNCTION AND THE CONSTRAINTS, AND IS CALLED FROM GRG.
C SUBROUTINE FREEN CALCULATES THE FREE ENERGY COEFFICIENTS (G/RT)
C USING THE DATA FROM GORDON AND MC.BRIDE.
C A SENSITIVITY ANALYSIS OF THE DUAL VARIABLES IS PERFORMED BY
C SUBROUTINE NPLUSK
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL*1 BLANK /' '/, SEMIC /';'/
C DIMENSION STATMENTS . LABELLED COMMONS ARE USED.
C N=NUMBER OF CHEMICAL SPECIES   +1
C M=NUMBER OF CHEMICAL ELEMENTS    (ATOMS) + 1
C K=NUMBER OF PHASES
C NN= NUMBER OF DIFFERENT TEMP. AND PRESSURE CONDITIONS.
C NE=1 IF THE FREE ENERGY COEFFICIENTS ARE ALREADY AVAILABLE. IF
C THEY ARE TO BE CALCULATED BY THE GORDON AND MC.BRIDE
C COEFFICIENTS, USE ANY OTHER INTEGER .
C NF=1 IF THE DUAL STARTING POINT ROUTINE IS TO BE USED.
C OTHERWISE, EQUAL NF TO ANY OTHER INTEGER,
C AND PROVIDE A DUAL STARTING POINT. DON'T FORGETTHAT
C THE FIRST DUAL VARIABLE IS DUMMY AND EQUAL TO 1.D0.
C A(I,J),I=1,M,J=1,N = EXPONENTS MATRIX. A(J,1) CORRESPONDS
C TO THE NORMALITY CONDITION.
C C(JJ,L),JJ=1,N, L=1,NN FREE ENERGY COEFFICIENTS OF SPECIES JJ
C T,P=EQUILIBRIUM TEMPERATURE AND PRESSURE
C X(I)=Z(I,1),I=1,M-1 =TRANSFORMED PRIMAL VARIABLES. FOR
C SUBROUTINE LIPSU2,X(J),J=1,N+1, CORRESPOND TO THE VARIABLES OF
C THE LINEAR  PROGRAM.
C XNEW(J),J=2,N =NUMBER OF MOLES OF SPECIES J; XNEW(1)=1.D0 IS A
C DUMMY VARIABLE CORRESPONDING TO THE NORMALITY CONDITION.
C XMF(J)=MOLAR FRACTION OF SPECIES J
C XLAG(I),I=1,K = LAGRANGE MULTIPLIER OF CONSTRAINT I
C S(J,L),J=1,N-1,L=1,7 =COEFFICIENTS FOR GORDON & MC.BRIDE
C POLYNOMIALS
C U(I,J) I=1,N-M, J=1,N IS THE MATRIX OF STOICHIOMETRIC COEFFICIE
C B(I),I=2,M = VECTOR OF AMOUNTS OF ELEMENT I.B(1)=1, DUMMY.
C F = OBJECTIVE FUNCTION.
      COMMON/AX/A(10,30)
      COMMON/BX/B(10)
      COMMON/CX/C(30,10)
      COMMON/DX/N,M/HX/K,NN/GX/F/XL/L
      COMMON/EX/T(30),P(30)
      COMMON/FX/X(31)/ZX/Z(10,5)
      DIMENSION XMF(30),VL(30),VH(30)
      COMMON/XNE/XNEW(31)/SX/SUMX/SUX/S(30,10)/UX/U(30,30)
      DIMENSION XLAG(10),SUMC(30)
C
C READ IN DATA
C
      READ(5,10)  N,M,K,NN,NE,NF
10    FORMAT (6I5)
C
C CALCULATE FREE ENERGY COEFFICIENTS BY GORDON AND MC.BRIDE
C
      IF (NE.EQ.1) GO TO 13
      SX =550.D0
C
C CALCULATE THE TEMPERATURE AT FIXED INTERVALS.
C
      DO 20 L=1,NN
      SX=SX +5.D1
      T(L)=SX
20    P(L)=1.D0
      N1=N-1
      READ (5,40) ((S(I,L),L=1,7),I=1,N1)
40    FORMAT (5G13.7)
      CALL FREEN
      GO TO 400
13    READ (5,311) (T(L),L=1,NN)
C
C IF THE C(I,L) MATRIX IS AVAILABLE BY OTHER SOURCES THAN
C GORDON & MC.BRIDE, READ NOW TEMPERATURE, PRESSURE, AND THE
C C(I,L) MATRIX.
C
      READ (5,311) (P(L),L=1,NN)
      READ (5,311) ((C(I,L),L=1,NN),I=1,N)
311   FORMAT (5F10.0)
C
C READ THE EXPONENT MATRIX AND THE B VECTOR.
C
400   READ (5,50) ((A(I,J),J=1,N),I=1,M)
50    FORMAT (13F4.0)
      READ (5,60) (B(I),I=1,M)
60    FORMAT (5F10.0)
C
C READ THE MATRIX OF STOICHIOMETRIC COEFFICIENTS.
C
      N2=N-M
      READ (5,70) ((U(I,J),J=1,N),I=1,N2)
70    FORMAT (10F4.0)
C
C PUT NF=1 IF YOU WANT THE PROGRAM TO CALCULATE A DUAL STARTING
C POINT. OTHERWISE, IT WILL NOW READ YOUR FIRST DUAL GUESS.
C
      IF (NF.EQ.1) GO TO 500
      READ (5,511) (XNEW(J),J=1,N)
511   FORMAT (5F10.0)
      TIME=SCLOCK(0.)
      GO TO 600
500   TIME=SCLOCK(0.)
C
C CALLS SUBROUTINE LIPSUB2.IT PROVIDES A STARTING POINT FOR THE
C DUAL PROBLEM.
      CALL LIPSU2
      XNEW(1)=1.D0
      DO 100 J=2,N
100   XNEW(J) = X(J - 1) + X(N)
      WRITE (6,12) (XNEW(J),J=1,N)
12    FORMAT (' FIRST DUAL POINT'/1X,6G16.8)
C
C SUBROUTINE SINGV WILL NOW TRANSFORM THE DUAL VARIABLES INTO
C PRIMAL ONES.
C
600   CALL SINGV
      M1=M-1
      WRITE (6,16) (Z(I,1),I=1,M1)
16    FORMAT (' FIRST PRIMAL STARTING POINT',4G16.8)
      DO 11 I=1,M1
11    X(I)=Z(I,1)
      CALL FTNCMD('ASSIGN 5=-DATA;')
      CALL FTNCMD ('ASSIGN 7=*SINK*;')
C OPTIMIZATION IS PERFORMED FOR NN DIFFERENT P AND T CONDITIONS
C A SCRATCH FILE IS CREATED TO WRITE DOWN THE DATA NEEDED FOR THE
C UBC SUBROUTINES GRGIN AND GRG2 TO PERFORM THE OPTIMIZATION
C
C FIRST ARE THE CONTROL CARDS
C
      IG=0.D0
      WRITE (5,15) M1,K,IG
15    FORMAT (3I6)
C
C NOW WRITE THE LOWER BOUNDS OF THE VARIABLES.WE ADD -30 TO THE
C PRIMAL STARTING POINT.
C
      WRITE (5,25)
25    FORMAT ('LBV=')
      DO 115 I=1,M1
115   VL(I)=X(I)-3.D1
      WRITE (5,35) (BLANK,I,VL(I),I=1,M1),SEMIC
35    FORMAT (6(A1,I3,G10.2))
C NOW WRITE THE UPPER BOUNDS OF THE VARIABLES.WE SET THEM AS 0.0
C
      WRITE (5,55)
55    FORMAT ('UBV=')
      DO 125 I=1,M1
125   VH(I)=0.D0
      WRITE (5,35) (BLANK,I,VH(I),I=1,M1),SEMIC
C
C PRIMAL CONSTRAINTS. IF THERE ARE MORE THAN ONE, THE FORMAT
C SHOULD BE : FORMAT('UBC= 1 1.D0 2 1.D0 K 1.D0 ;')
C
      WRITE (5,66)
```

```
66    FORMAT('UBC= 1 1.D0 ;')
      WRITE (5,65)
65    FORMAT ('QUAD')
C
C EPNEWT IS THE TOLERANCE FOR THE PRIMAL CONSTRAINTS.THE ACCURACY
C OF THE RESULTS IS VERY SENSITIVE TO THIS VALUE.
C
      WRITE (5,75)
75    FORMAT ('EPNEWT=1.D-10')
      WRITE (5,85)
C
C EPSTOP IS THE TOLERANCE FOR THE OBJECTIVE FUNCTION
C
85    FORMAT ('EPSTOP=1.D-6')
      WRITE (5,95)
95    FORMAT ('EPSBOUND=1.D-6')
      WRITE (5,105)
105   FORMAT ('EPSPIV=1.D-6')
      WRITE (5,61)
C
C PRINTCTL MAY BE SET AT HIGHER VALUES (UP TO 4) FOR MORE
C INFORMATION ON THE PROGRESS OF THE OPTIMIZATION.
C
61    FORMAT (' PRINTCTL=1')
      WRITE (5,107)
107   FORMAT('X=')
      WRITE (5,109) (X(I),I=1,M1)
109   FORMAT (6G18.6)
      WRITE (5,108)
108   FORMAT ('OPTIMIZE'/'GO'/'STOP')
      REWIND 5
C NOW GRGIN CAN READ INPUT FROM  -DATA;GRG2 AND GRGRES WILL WRITE
C THEIR OUTPUTS INTO THE SCRATCH FILE -GRGOUT
      CALL FTNCMD('ASSIGN 6=-GRGOUT;')
      CALL GRGIN(&1,&2)
      CALL GRG2(X,F,&1)
      CALL GRGRES
      GO TO 2
1     WRITE (7,3)
3     FORMAT (' GRG HAS FAILED')
      GO TO 202
2     REWIND 6
C
C SUBROUTINE SKIP READS THE LAGRANGE MULTIPLIERS FOR EACH PRIMAL
C CONSTRAINT. THEY ARE THE NEGATIVE OF THE TOTAL NUMBER OF
C MOLES IN EACH CONSTRAINT.
C
      CALL SKIP(&202)
      READ (6,26) (XLAG(I),I=1,K)
26    FORMAT (1X,10F13.0)
      WRITE (7,205) T(L),P(L),F
205   FORMAT ('T(K)=',F10.0,'P(AT)=',F10.0,'OBJ.F=',G16.8)
      WRITE (7,215)
215   FORMAT (' MOLAR FRACTIONS')
```

```
      DO 33 J=2,N
      SUMC(J)=0.D0
      DO 31 I=1,M1
      II=I+1
31    SUMC(J)=SUMC(J)+A(II,J)*X(I)
      IF (SUMC(J).GE.0.D0) GO TO 34
      R=DEXP(SUMC(J)-C(J,L))
      GO TO 33
34    R=0.D0
33    XMF(J)=R
      WRITE (7,235) (XMF(J),J=2,N)
235   FORMAT (1X,6G16.8)
      WRITE (7,238) (-XLAG(I),I=1,K)
238   FORMAT (' TOTAL NUMBER OF MOLES',G16.8)
      WRITE (7,239)
239   FORMAT ('NUMBER OF MOLES')
      DO 237 J=2,N
237   XNEW(J)=-XLAG(1)*XMF(J)
      WRITE (7,240) (XNEW(J),J=2,N)
240   FORMAT (1X,6G16.8)
      XNEW(1)=1.D0
      SUMX=-XLAG(1)
      CALL NPLUSK
      GO TO 23
23    TIME=SCLOCK(TIME)
      WRITE (7,19) TIME
19    FORMAT (' EXECUTION TIME =',F6.4)
      STOP
      END
C
C SUBROUTINE LIPSU2 CALCULATES A DUAL STARTING POINT APPROXIMATIN
C THE DUAL PROBLEM TO A LINEAR PROGRAM.THE LINEAR PROGRAM IS
C SOLVED BY SUBROUTINE LIPSUB FROM UBC.
C
      SUBROUTINE LIPSU2
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TABLO(30,30),NVIN(30),NVOUT(30),BBOBJ(20),
     1        UBOBJ(20),BRHS(20),UBRHS(10)
      COMMON/AX/A(10,30)
      COMMON/BX/B(10)
      COMMON/CX/C(30,10)/GX/OPTIM
      COMMON/DX/NVARS,NCONST
      COMMON/FX/X(31)
      N1=NVARS-1
      M1=NCONST-1
      NEQUAL=M1
      MAXIM=0
      IFOBJ=0
      IFRHS=0
      NCOLS=NVARS+1
      NROWS=NCONST+1
C ZERO THE TABLEAU
      DO 200 J=1,NCOLS
      DO 200 I=1,NROWS
```

```
200    TABLO(I,J)=0.D0
C FIRST ROW IN THE TABLEAU IS THE OBJECTIVE FUNCTION
       SUM=0.D0
       DO 101 J=1,N1
       JJ=J+1
       TABLO(1,J)=C(JJ,1)
101    SUM=SUM+TABLO(1,J)
       TABLO(1,NVARS)=SUM
       TABLO(2,NVARS)=-1.D0
       TABLO(2,NCOLS)=-.0001D0
       DO 203 I=2,NCONST
       DO 202 J=1,N1
202    TABLO(I + 1, J) = A(I, J + 1)
       TABLO(I + 1, NVARS) = 1.D0
203    TABLO(I + 1, NCOLS) = B(I)
260    NLVARS = NVARS
       CALL LIPSUB(TABLO, 30, NCONST, NLVARS, NEQUAL, MAXIM,
      1           IFOBJ, IFRHS, TOL, NVIN, NVOUT,BBOBJ,
      2            UBOBJ, BBRHS, UBRHS, &540)
       NP1=NLVARS+1
       OPTIM =TABLO(1,NP1)
       IF (MAXIM.NE.1) OPTIM=-OPTIM
       DO 12 I=1,NVARS
12     X(I)=0.D0
       DO 13 I=2,NROWS
       J = NVIN(I)
13     X(J)=TABLO(I,NP1)
       WRITE (6,320)   (X(J),J=1,NVARS)
320    FORMAT (1X,5G18.6)
       RETURN
540    WRITE (6,560)
560    FORMAT ('0DUAL STARTING POINT ROUTINE FAILED')
       DO 580 I = 1, NVARS
580    X(I) = 10.0D0
       RETURN
       END
C
C SUBROUTINE GCOMP CALCULATES THE VALUES OF THE OBJECTIVE FUNCTIO
C AND OF THE CONSTRAINTS
C
       SUBROUTINE GCOMP(G,X)
       IMPLICIT REAL*8 (A-H,O-Z)
       COMMON/AX/A(10,30)/BX/B(10)/CX/C(30,10)/DX/N,M/XL/L
       COMMON/HX/K,NN
       DIMENSION G(1),X(1),SUMF(5),XF(30),XMF(30)
       DO 10 I=1,K
       SUMF(I)=0.D0
       DO 20 J=2,N
       XF(J)=0.D0
       DO 30 II=2,M
       I1=II-1
30     XF(J)=XF(J)+A(II,J)*X(I1)
       IF (XF(J).GE.0.D0)  GO TO 40
       XMF(J)=XF(J)-C(J,L)
       R=DEXP(XMF(J))
       GO TO 20
40    R=0.D0
20    SUMF(I)=SUMF(I)+R
10    G(I)=SUMF(I)
       SUM=0.D0
       DO 50 J=2,M
       JJ=J-1
50    SUM=SUM+B(J)*X(JJ)
       G(K+1) = -SUM
       RETURN
       END
C
C SUBROUTINE SINGV CALCULATES A FIRST STARTING POINT FOR THE PRIM
C PROBLEM.IT SOLVES AN OVERDETERMINED SYSTEM OF EQUATIONS THAT
C RELATES PRIMAL AND DUAL VARIABLES.
C THE METHOD USED IS A SINGULAR VALUE DECOMPOSITION.
C
       SUBROUTINE SINGV
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION  A(30,30),V(30,30),S(30)
       COMMON/AX/AD(10,30)/CX/C(30,10)/XNE/XNEW(31)/DX/N1,M1
       COMMON/ZX/X(10,5)/HX/K,NN
       NP=1.D0
       N=N1-1
       M=M1-1
       MNP=M+NP
       NDIMAU=30
       NDIMV=30
       DO 10 I=1,N
       II=I+1
       DO 10 J=1,M
       JJ=J+1
10    A(I,J)=AD(JJ,II)
       SUMX=0.D0
       DO 12 J=2,N1
12    SUMX=SUMX+XNEW(J)
       DO 11 I=1,N
       II=I+1
       DO 11 J=M1,MNP
       IF (XNEW(II).LE.0.D0) XNEW(II)=1.D-16
11    A(I,J)=C(II,1)+DLOG(XNEW(II))-DLOG(SUMX)
       WRITE (6,14) ((A(I,J),J=1,MNP),I=1,N)
14    FORMAT (1X,5G18.6)
       CALL DSLSVD(A,S,V,NDIMAU,NDIMV,N,M,NP,&140)
       EPS=1.D-6
       SS=S(1)*EPS
       DO 60 J=1,M
       IF (S(J).LT.SS) GO TO 70
       DO 50 I=1,M
50    V(I,J)=V(I,J)/S(J)
60    CONTINUE
       J=M+1
       GO TO 90
```

```
70     WRITE (6,80) ((V(I,K),I=1,M),K=J,M)
80     FORMAT (1X,4G13.5)
90     IF (J.GT.M) GO TO 120
       DO 110 K=J,M
       DO 110 I=1,M
110    V(I,K)=0.D0
120    MP1=M+1
       CALL DGMULT(V,A(1,MP1),X,M,M,NP,NDIMV,NDIMAU,10)
       RETURN
140    WRITE (6,150)
150    FORMAT (' ERROR RETURN FROM DSLSVD')
       STOP
       END
C
C SUBROUTINE SKIP WILL READ THE LAGRANGIAN MULTIPLIERS FOR
C THE PRIMAL CONSTRAINTS AS PRINTED BY GRGEG
C IN  THE SCRATCH FILE -GRGOUT.
C
       SUBROUTINE SKIP(*)
       LOGICAL*1 RECORD(150)
       INTEGER*2 LEN
       LOGICAL EQCMP
100    CALL READ(RECORD,LEN,0,LNR,6,&200)
       IF (LEN .LT. 70 .OR. LEN .GT. 75) GO TO 100
       IF (EQCMP (21, RECORD, '0LAGRANGE MULTIPLIERS')) RETURN
       GO TO 100
200    RETURN 1
       END
C
C SUBROUTINE FREEN CALCULATES THE FREE ENERGY PARAMETERS FROM
C THE GORDON AND MC BRIDE COEFFICIENTS.
C
       SUBROUTINE FREEN
       IMPLICIT REAL*8 (A-H,O-Z)
       COMMON/CX/C(30,10)/SUX/S(30,10)
       COMMON/DX/N,M/HX/K,NN
       COMMON/EX/T(30),P(30)
       DO 10 L=1,NN
       C(1,L)=0.D0
       N1=N-1
       DO 10 J=1,N1
       JJ=J+1
10     C(JJ,L)=S(J,1)*(1.D0-DLOG(T(L)))-S(J,2)*T(L)/2.D0-
      1     (S(J,3)*T(L)**2.D0)/6.D0 -(S(J,4)*T(L)**3.D0)/1.2D1
      2    -(S(J,5)*T(L)**4.D0)/2.D1+S(J,6)/T(L)-S(J,7)
      3    +DLOG(P(L))
       WRITE(6,12)
12     FORMAT ('MATRIX OF FREE ENERGY COEFFICIENTS')
       WRITE(6,11) ((C(J,L),J=1,N),L=1,NN)
11     FORMAT (1X,4F12.7)
       RETURN
       END
C
C SUBROUTINE NPLUSK PERFORMS SENSITIVITY ANALYSIS SOLVING A SYSTE
```

```
C SYSTEM OF LINEAR EQUATIONS WITH SUBROUTINE DSLIMP(FROM UBC).THE
C U(I,J) THAT IS , THE STOICHIOMETRIC COEFFICIENTS FOR THE
C REACTIONS THAT TAKE PLACE,  ARE PROVIDED BY THE USER.
C
      SUBROUTINE NPLUSK
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/AX/A(10,30)/BX/B(10)/XNE/DELT(31)/UX/U(30,30)
      COMMON/CX/C(30,10)/DX/N1,M/HX/K,NN/EX/T(30),P(30)/SX/SUMX
      DIMENSION DA(50,50), DT(50,50), DB(50), DX(50),
     1         DRZ(50), IPERM(100)
      DIMENSION SUM(30),V(30),SUMB(30)
      N=N1+K
      NDIMAT=50
      N2=N1-M
      DEPS=1.D-8
      ITMAX=0
C NOW WE NEED TO FORM  THE (N1+KXN1+K) MATRIX
      DO 83 I=1,N
      DO 83 J=1,N
83    DA(I,J)=0.D0
C NORMALITY AND ORTHOGONALITY CONDITIONS
      DA(1,1)=1.D0
      DO 84 I=2,M
      DO 84 J=2,N1
      DA(I,1)=-B(I)
84    DA(I,J)=A(I,J)
C NOW, SUMMATION OF NUMBER OF MOLES
      DO 4 I=1,K
      LL=I+M
      DO 5 J=2,N1
5     DA(LL,J)=1.D0
4     DA(LL,N)=-1.D0
C NOW, EQUILIBRIUM CONDITIONS
      DO 8 I=1,N2
      SUMB(I)=0.D0
      DO 7 J=1,N1
      LL=M+K+I
      DA(LL,J)=U(I,J)/DELT(J)
7     SUMB(I)=SUMB(I)+U(I,J)
8     DA(LL,N)=SUMB(I)/SUMX
C NOW CALCULATE THE RIGHT HAND SIDE VECTOR
      DO 95  L=2,NN
      M3=  M+K
      DO 10 I=1,M3
10    DB(I)=0.D0
      DO 11 I=1,N2
      SUM(I)=0.D0
      DO 9 J=1,N1
9     SUM(I)=SUM(I)-U(I,J)*(C(J,L)-C(J,1))
      LL=I+M+K
11    DB(LL)=SUM(I)
      NRHS=L-1
C SOLVE THE SYSTEM
      CALL DSLIMP(DA,DT,DB,DX,DRZ,IPERM,N,NDIMAT,DEPS,NRHS,ITMAX)
```

```
C WRITE OUT RESULTS
      WRITE (7,900)
900   FORMAT (' VARIATION OF NUMBER OF MOLES')
      WRITE (7,110)  (DX(I),I=1,N)
110   FORMAT (1X,6G14.6)
C CALCULATE THE NEW NUMBER OF MOLES
      DO 15 I=1,N
15    V(I)=DX(I)+DELT(I)
      W=P(L)
      Z=T(L)
      WRITE (7,120) W,Z
120   FORMAT (' P=',F10.0,' T(KELVIN)=',F10.0,1X,'SOLUTION')
      WRITE (7,130) (V(I),I=1,N)
130   FORMAT (1X,6G16.8)
95    CONTINUE
      RETURN
      END
```

Program COMP3

```
C
C
C
C THIS PROGRAM SOLVES THE DUAL G.P. FOR THE CHEMICAL EQUILIBRIUM
C PROBLEM. IT CALLS THE SUBROUTINE LIPSU2 TO GET A FIRST POINT
C FOR THE OPTIMIZATION. THEN IT CALLS SUBROUTINES GRGIN AND GRG2
C WHICH SOLVE THE PROBLEM ITSELF.SUBROUTINE GCOMP CALCULATES THE
C OBJECTIVE FUNCTION AND THE CONSTRAINTS, AND IS CALLED FROM
C SUBROUTINE GRG2.SUBROUTINES GRGIN AND GRG2 ARE FROM UBC.
C THE PROCEDURE IS REPEATED FOR DIFFERENT TEMPERATURES AND
C PRESSURES.
C SUBROUTINE FREEN CALCULATES THE FREE ENERGY COEFFICIENTS (G/RT)
C USING THE DATA FROM GORDON & MC.BRIDE
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL*1 BLANK /' '/, SEMIC /';'/
C DIMENSION STATMENTS . LABELLED COMMONS ARE USED.
C N=NUMBER OF CHEMICAL SPECIES +1
C M=NUMBER OF CHEMICAL ELEMENTS (ATOMS) + 1
C K=NUMBER OF PHASES
C NN= NUMBER OF DIFFERENT TEMP. AND PRESSURE CONDITIONS.
C NE=1 IF THE FREE ENERGY COEFFICIENTS AT TEMPERATURE T ARE
C AVAILABLE. IF THEY HAVE TO BE CALCULATED BY
C THE GORDON AND MC.BRIDE POLYNOMIALS, SET NE EQUAL TO
C ANY INTEGER DIFFERENT THAN ONE.
C NF=1 IF THE PROGRAM HAS TO PROVIDE A FIRST STARTING POINT.
C IF YOU PROVIDE THE FIRST STARTING POINT, SET NF EQUAL TO
C ANY OTHER INTEGER.
C A(I,J),I=1,M,J=1,N = EXPONENTS MATRIX.A(1,1)=1 CORRESPONDS
C TO THE NORMALITY CONDITION.
C C(J,L), J=1,N, L=1,NN  = FREE ENERGY COEFFICIENTS OF SPECIES J
C AT T(L),P(L).SET C(1,L)=0.
C T,P=EQUILIBRIUM TEMPERATURE AND PRESSURE.
C XNEW(J),J=2,N = NUMBER OF MOLES OF SPECIES J AT EQUILIBRIUM.
C FOR F=1, XNEW =1 (DUMMY SPECIES CORRESPONDING TO THE NORMALITY
C CONDITION)
C X(J),J=1,N  = FOR SUBROUTINE LIPSU2, CORRESPONDS TO THE LINEAR
C PROGRAM VARIABLES. FOR GRG, DUAL VARIABLES, EQUAL TO XNEW(J).
C XMF(J),J=2,N   MOLAR FRACTION OF SPECIES J
C S(J,L) J=1,N-1, L=1,7 = COEFFICIENTS FOR GORDON AND MC.BRIDE
C POLYNOMIALS FOR SPECIES J+1.
C B(I),I=2,M VECTOR OF AMOUNTS OF ELEMENT I, THAT ARE CONSERVED.
C B(1)=1, DUMMY.
C F= OBJECTIVE FUNCTION.
C
C
      COMMON/AX/A(10,30)
      COMMON/BX/B(10)
      COMMON/CX/C(30,10)
      COMMON/DX/N,M/HX/K,NN/GX/F/XL/L
      COMMON/EX/T(30),P(30)/SX/S(30,10)
      COMMON/FX/X(31)
      DIMENSION XMF(30),H(30),Q(30)
      DIMENSION XNEW(31)
C
C READ IN DATA
C
      READ(5,10)  N,M,K,NN,NE,NF
10    FORMAT (6I5)
C
C CALCULATE FREE ENERGY COEFFICIENTS BY GORDON AND MC.BRIDE
C
      IF (NE.EQ.1) GO TO 13
C
C CALCULATE THE TEMPERATURE AT FIXED INTERVALS.
C
      SX=55.D1
      DO 20 L=1,NN
      SX=SX+50.D0
      T(L)=SX
20    P(L)=1.D0
      N1=N-1
      READ (5,40) ((S(I,L),L=1,7),I=1,N1)
40    FORMAT (5G13.7)
      CALL FREEN
      GO TO 400
C
C IF THE C(I,L) MATRIX IS AVAILABLE BY OTHER SOURCES OTHER
C THAN GORDON AND MC.BRIDE, READ TEMPERATURES, PRESSURES, &
C THE C(I,L) MATRIX.
C
13    READ (5,311) (T(L),L=1,NN)
      READ (5,311) (P(L),L=1,NN)
      READ (5,311) ((C(I,L),L=1,NN),I=1,N)
311   FORMAT (5F10.0)
C
C READ THE EXPONENT MATRIX AND THE B VECTOR
C
400   READ (5,50) ((A(I,J),J=1,N),I=1,M)
50    FORMAT (13F4.0)
      READ (5,60) (B(I),I=1,M)
60    FORMAT (5F10.0)
      IF (NF.EQ.1) GO TO 500
C
C IF YOU HAVE A GOOD STARTING POINT, THE PROGRAM SHOULD READ IT N
C OTHERWISE, IT CALCULATES ITS OWN STARTING POINT.
C
      READ (5,510) (XNEW(J),J=1,N)
510   FORMAT (5F10.0)
      TIME=SCLOCK(0.)
      GO TO 600
500   TIME=SCLOCK(0.)
C CALLS SUBROUTINE LIPSUB2.IT PROVIDES A STARTING POINT FOR THE
C OPTIMIZATION ROUTINE.
      CALL LIPSU2
      XNEW(1)=1.D0
      DO 100 J=2,N
100   XNEW(J) = X(J - 1) + X(N)
      WRITE (6,12) (XNEW(J),J=1,N)
12    FORMAT (' FIRST STARTING POINT'/1X,6G16.8)
      DO 200 L=1,NN
C OPTIMIZATION IS PERFORMED FOR NN DIFFERENT P AND T CONDITIONS
C A SCRATCH FILE IS CREATED TO WRITE DOWN THE DATA NEEDED FOR THE
C UBC SUBROUTINES GRGIN AND GRG2 TO PERFORM THE OPTIMIZATION
      CALL FTNCMD('ASSIGN 5=-DATA;')
C
C FIRST COME THE CONTROL CARDS.
C
      WRITE (5,15) N,M,M
15    FORMAT (3I6)
C
C NOW WRITE THE LOWER BOUNDS FOR THE VARIABLES.
C
      WRITE (5,25)
25    FORMAT ('LBV=')
      DO 115 I=1,N
115   H(I)=1.D-60
      WRITE (5,35) (BLANK,I,H(I),I=1,N),SEMIC
35    FORMAT (6(A1,I3,G10.2))
C
C NOW, THE UPPER BOUND FOR THE VARIABLES.
C
      WRITE (5,55)
55    FORMAT ('UBV=')
      DO 125 I=1,N
125   Q(I)=40.D0
      WRITE (5,35) (BLANK,I,Q(I),I=1,N),SEMIC
      WRITE (5,65)
65    FORMAT ('QUAD')
C
C EPNEWT IS THE TOLERANCE FOR THE CONSTRAINTS.
C
      WRITE (5,75)
75    FORMAT ('EPNEWT=1.D-6')
C
C EPSTOP IS THE TOLERANCE FOR THE OBJECTIVE FUNCTION.
C
      WRITE (5,85)
85    FORMAT ('EPSTOP=1.D-6')
      WRITE (5,95)
95    FORMAT ('EPSBOUND=1.D-6')
C
C EPSPIV SHOULD BE OF THE ORDER OF THE LOWER BOUNDARY.
C
      WRITE (5,105)
105   FORMAT ('EPSPIV=1.D-16')
C
C PRINTCTL MAY BE SET AT HIGHER VALUES (UP TO 4) FOR MORE
C INFORMATION ON THE PROGRESS OF THE OPTIMIZATION. IF 'NOECHO'
C IS WRITEN ON FILE -DATA JUST BEFORE THE CONTROL CARDS, AND
```

```
C IF PRINTCTL =0, NO OUTPUT IS WRITTEN FROM GRG.
C
      WRITE (5,61)
61    FORMAT (' PRINTCTL=1')
      DO 101 I=1,N
101   X(I)=XNEW(I)
      WRITE (5,107)
107   FORMAT('X=')
      WRITE (5,109) (X(I),I=1,N)
109   FORMAT (6G18.6)
      WRITE (5,108)
108   FORMAT ('OPTIMIZE'/'GO'/'STOP')
      REWIND 5
C NOW GRGIN CAN READ INPUT FROM  -DATA;GRG WILL WRITE ITS OUTPUT
C INTO THE SCRATCH FILE -GRGOUT.
      CALL GRGIN(&1,&2)
      CALL GRG2(XNEW,F,&1)
      GO TO 2
1     WRITE (6,3)
3     FORMAT (' GRG HAS FAILED')
2     WRITE (6,205) T(L),P(L),F
205   FORMAT ('T(K)=',F10.0,'P(AT)=',F10.0,'OBJ.F=',G16.8)
      WRITE (6,215)
215   FORMAT ('NUMBER OF MOLES')
      WRITE (6,225)(XNEW(I),I=1,N)
225   FORMAT (1X,6G16.8)
      SUM=0.D0
      DO 8 J=2,N
8     SUM=SUM+XNEW(J)
      WRITE (6,235) SUM
235   FORMAT (' TOTAL NUMBER OF MOLES=',G16.8)
      DO 9 J=2,N
9     XMF(J)=XNEW(J)/SUM
      WRITE (6,245) (XMF(J),J=2,N)
245   FORMAT (1X,' MOLAR FRACTIONS',1X,6G16.8)
200   CONTINUE
      GO TO 23
23    TIME=SCLOCK(TIME)
      WRITE (6,19) TIME
19    FORMAT (' EXECUTION TIME =',F6.4)
      STOP
      END
C
C SUBROUTINE LIPSU2 CALCULATES A STARTING POINT APPROXIMATING THE
C DUAL PROBLEM TO A LINEAR PROGRAM. THE LINEAR PROGRAM IS SOLVED
C BY SUBROUTINE LIPSUB FROM UBC.
C
      SUBROUTINE LIPSU2
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION TABLO(30,30),NVIN(30),NVOUT(30),BBOBJ(20),
     1        UBOBJ(20),BRHS(20),UBRHS(10)
      COMMON/AX/A(10,30)
      COMMON/BX/B(10)
      COMMON/CX/C(30,10)/GX/OPTIM
      COMMON/DX/NVARS,NCONST
      COMMON/FX/X(31)
      N1=NVARS-1
      M1=NCONST-1
      NEQUAL=M1
      MAXIM=0
      IFOBJ=0
      IFRHS=0
      NCOLS=NVARS+1
      NROWS=NCONST+1
C ZERO THE TABLEAU
      DO 200 J=1,NCOLS
      DO 200 I=1,NROWS
200   TABLO(I,J)=0.D0
C FIRST ROW IN THE TABLEAU IS THE OBJECTIVE FUNCTION
      SUM=0.D0
      DO 101 J=1,N1
      JJ=J+1
      TABLO(1,J)=C(JJ,1)
101   SUM=SUM+TABLO(1,J)
      TABLO(1,NVARS)=SUM
      TABLO(2,NVARS)=-1.D0
      TABLO(2,NCOLS)=-.0001D0
      DO 203 I=2,NCONST
      DO 202 J=1,N1
202   TABLO(I + 1, J) = A(I, J + 1)
      TABLO(I + 1, NVARS) = 1.D0
203   TABLO(I + 1, NCOLS) = B(I)
260   NLVARS = NVARS
      CALL LIPSUB(TABLO, 30, NCONST, NLVARS, NEQUAL, MAXIM,
     1           IFOBJ, IFRHS, TOL, NVIN, NVOUT,BBOBJ,
     2            UBOBJ, BBRHS, UBRHS, &540)
      NP1=NLVARS+1
      OPTIM =TABLO(1,NP1)
      IF (MAXIM.NE.1) OPTIM=-OPTIM
      DO 12 I=1,NVARS
12    X(I)=0.D0
      DO 13 I=2,NROWS
      J = NVIN(I)
13    X(J)=TABLO(I,NP1)
      WRITE (6,320)   (X(J),J=1,NVARS)
320   FORMAT (1X,5G18.6)
      RETURN
540   WRITE (6,560)
560   FORMAT (' STARTING POINT ROUTINE FAILED')
      DO 580 I = 1, NVARS
580   X(I) = 10.0D0
      RETURN
      END
C
C SUBROUTINE GCOMP CALCULATES THE VALUES OF THE OBJECTIVE
C FUNCTION AND OF THE CONSTRAINTS.
C
      SUBROUTINE GCOMP(G,X)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/AX/A(10,30)/BX/B(10)/CX/C(30,10)/DX/N,M/XL/L
      DIMENSION G(1),X(1),SUM(30)
      DO 20 I=1,M
      SUM(I)=0.D0
      DO 10 J=1,N
      IF (A(I,J) .LE. 0.D0) X(J)=0.D0
10    SUM(I)=SUM(I)+A(I,J)*X(J)
20    G(I)=SUM(I)-B(I)
      M1=M+1
      DELT=0.D0
      DO 30 I=2,N
30    DELT=DELT+X(I)
      SUMA=0.D0
      DO 40 I=1,N
      DLX = -1.E50
      IF (X(I) .GT. 0.0D0) DLX = DLOG(X(I))
40    SUMA=SUMA+X(I)*(DLX + C(I,L))
      DLD = -1.E40
      IF (DELT .GT. 0.0D0) DLD = DLOG(DELT)
      G(M1)=SUMA-DELT*DLD
      RETURN
      END
C
C SUBROUTINE FREEN CALCULATES FREE ENERGY PARAMETERS
C
      SUBROUTINE FREEN
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON/CX/C(30,10)/SX/S(30,10)/DX/N,M/HX/K,NN
      COMMON/EX/T(30),P(30)
      DO 10 L=1,NN
      C(1,L)=0.D0
      N1=N-1
      DO 10 J=1,N1
      JJ=J+1
10    C(JJ,L)=S(J,1)*(1.D0-DLOG(T(L)))-S(J,2)*T(L)/2.D0-
     1       (S(J,3)*T(L)**2.D0)/6.D0-(S(J,4)*T(L)**3.D0)/1.2D1
     2      -(S(J,5)*T(L)**4.D0)/2.D1+S(J,6)/T(L)-S(J,7)
     3      +DLOG(P(L))
      WRITE (6,7)
7     FORMAT (' MATRIX OF FREE ENERGY COEFFICIENTS')
      WRITE(6,11) ((C(J,L),J=1,N),L=1,NN)
11    FORMAT (1X,4F12.7)
      RETURN
      END
```

APPENDIX B: EXAMPLES

PROBLEM 1: HYDRAZINE COMBUSTION Passy and Wilde,1968

Species = 10 Elements = 3 Phases = 1

G/RT = 47.8907 (1) Temperature= 3500 K
G/RT = 47.8907 (2) Pressure= 51.2 atm.

i	Elements	Bi	Zi (1)	Zi (2)
1	H	2.0	-9.78896	-9.78903
2	O	1.0	-15.2128	-15.2127
3	N	1.0	-13.1000	-13.1000

k	j	Species	Cj	σ j (1)	σ j (2)
1	2	H	-6.089	4.06 E-2	3.78 E-2
	3	H2	-17.164	1.48 E-1	1.47 E-1
	4	H2O	-34.054	7.83 E-1	7.85 E-1
	5	N	-5.914	1.41 E-3	1.24 E-3
	6	N2	-24.721	4.85 E-1	4.86 E-1
	7	NH	-14.986	6.93 E-4	6.06 E-4
	8	NO	-24.100	2.72 E-2	2.43 E-2
	9	O	-10.708	1.79 E-2	1.81 E-2
	10	O2	-26.662	3.73 E-2	3.80 E-2
	11	OH	-22.179	9.69 E-2	9.74 E-2

λ Total number of moles = 1.638 (1)
λ Total number of moles = 1.63799 (2)
Summation of molar fractions at optimum (1) = 0.995586
Summation of molar fractions at optimum (2) = 1.000001

NOTES:
(1) = Passy and Wilde, 1968
(2) = This thesis

PROBLEM 2 : METHANE AND WATER REACTION Perry, 1968

Species = 5 Elements = 3 Phases = 1

G/RT = 79.3605 (1)
G/RT = 79.3605 (2)

Temperature = 1095 K
Pressure = 1 atm.

i	Elements	Bi	Zi(1)	Zi(2)
1	C	2.0	-24.9739	-24.9738
2	H	14.0	-0.7922	-0.7922
3	O	3.0	-0.2007	-0.2007

k	j	Species	Cj	б j(1)	б j(2
1	2	CO	-24.025	1.5174	1.5174
	3	CO2	-47.413	.3107	.3107
	4	H2O	-23.067	.8612	.8613
	5	H2	0.0	5.7942	5.7943
	6	CH4	2.0847	.1722	.1722

λ Total number of moles = 8.6558 (1)
λ Total number of moles = 8.6559 (2)
Summation of molar fractions at optimum = 1.0000 (1)
Summation of molar fractions at optimum = 1.0000 (2)

Notes:
(1)= Perry,1968
(2)= This thesis

PROBLEM 3 : WATER GAS REACTION Dinkel and Lakshmanan,1975

Species = 4 Elements = 3 Phases = 1

G/RT = 90.4788 (1)
G/RT = 90.4789 (2)

Temperature = 1000 K
Pressure = 1 atm.

i	Elements	Bi	Zi (1)	Zi (2)
1	C	1.0	N.A	-5.31206
2	O	2.0	N.A	-33.4888
3	H	2.0	N.A	-9.09497

k	j	Species	Cj	б j (1)	б j (2)
1	2	CO	-37.4239	.505025	.505044
	3	H2O	-50.3023	.505025	.505088
	4	CO2	-70.8924	.494975	.494978
	5	H2	-16.7936	.494975	.494978

λ Total number of moles = 3.000000 (1)
λ Total number of moles = 3.000088 (2)
Summation of molar fractions at optimum = 1.000000 (1)
Summation of molar fractions at optimum = 1.000031 (2)

Notes:
(1) = Dinkel and Lakshmanan, 1975
(2) = This thesis.

PROBLEM 4 : RESPIRATORY SYSTEM Dembo, 1976

Species =31 Elements =12 Phases = 3

G/RT = (1) Temperature = N.A.
G/RT = 90.4789 (2) Pressure = N.A.

This problem was solved with the untransformed primal variables t, scaled as in the reference.

i	Elements	Bi	scaled ti (1)	scaled ti
1	O2	.0013317	2.517968	2.508494
2	CO2	.0022709	2.539194	2.525456
3	N2	.0024855	7.657035	7.664300
4	H+	4.67	1.221926	1.193585
5	OH-	4.671973	7.467072	7.645168
6	Cl-	.008140	1.291600	1.314099
7	Na+	.008092	4.283088	4.384049
8	K+	.005	2.781697	2.651293
9	HBl-	.000909	1.787073	1.761015
10	HPp-	.00088	2.001670	2.017308
11	HPr-	.00119	6.496106	6.487580
12	Z(charge)	0	6.449689	6.408977

k	j	Species	Cj	δ j (1)	δ j(2)
1	2	O2	-10.89	4.46 E-3	4.40 E-3
	3	CO2	- 7.69	1.83 E-3	1.81 E-3
	4	N2	-11.49	2.47 E-2	2.45 E-2
	5	H2O	-36.44	2.02 E-3	2.00 E-3
2	6	O2	0	7.32 E-5	7.05 E-5
	7	CO2	0	7.37 E-4	7.10 E-4
	8	N2	0	2.22 E-4	2.15 E-4
	9	H+	0	2.24 E-8	3.35 E-8
	10	OH-	0	3.43 E-7	2.15 E-7
	11	CL-	0	5.83 E-2	5.69 E-2
	12	Na+	0	8.09 E-2	1.23 E-1
	13	H2O	-39.23	2.89 E+1	2.80 E+1
	14	HCO3-	-21.20	1.40 E-2	8.74 E-3
	15	H2CO3-	0	6.72 E-21	6.48 E-21
	16	CO3-	6.25	2.18 E-5	8.80 E-6
	17	HPp-	0	8.75 E-3	5.67 E-3
3	18	O2	0	4.52 E-5	4.73 E-5
	19	CO2	0	4.55 E-4	4.76 E-4
	20	N2	0	1.37 E-4	1.44 E-4
	21	H+	0	2.13 E-8	2.25 E-8

PROBLEM 4 : RESPIRATORY SYSTEM Continued

k	j	Species	Cj	δ j (1)	δ j (2)
3	22	OH-	0	1.38 E-7	1.44 E-7
	23	Cl-	0	2.34 E-2	2.48 E-2
	24	K+	0	5.00 E-2	5.00 E-2
	25	H2O	-39.23	1.78 E+1	1.87 E+1
	26	HCO3-	-21.20	5.64 E-3	5.86 E-3
	27	H2CO3	0	4.15 E-21	4.35 E-21
	28	CO3-	6.25	5.69 E-6	5.89 E-6
	29	HBl	0	3.10 E-3	3.31 E-3
	30	HBlO2-	-16.23	8.74 E-3	9.34 E-3
	31	HPr-	0	1.20 E-2	1.22 E-2

λ_1 Total number of moles, phase 1 = N.A. (1)
λ_2 Total number of moles, phase 2 = N.A. (1)
λ_3 Total number of moles, phase 3 = N.A. (1)
λ_1 Total number of moles, phase 1 = .032700 (2)
λ_2 Total number of moles, phase 2 = 28.1013 (2)
λ_3 Total number of moles, phase 3 = 15.8586 (2)
Summation of molar fractions, phase 1 = 1.000000 (2)
Summation of molar fractions, phase 2 = 1.000014 (2)
Summation of molar fractions, phase 3 = .999991 (2)

Notes:
(1) = Dembo, 1976
(2) = This thesis

PROBLEM 5 : COMBUSTION OF PROPANE (1)

Species =10 Elements = 4 Phases = 1

G/RT = 777.6387 (1) Temperature = 2200 K
G/RT = 777.6387 (2) Pressure = 40 atm.

i	Elements	Bi	Zi (1)	Zi (2)
1	H	8.0	N.A.	-11.4153
2	C	3.0	N.A.	-20.1267
3	O	10.0	N.A.	-15.8052
4	N	40.0	N.A.	-11.6971

k	j	Species	Cj	δ j (1)	δ j (2)
1	2	H2	-15.6191	.0200735	.0204434
	3	H	-.7824	.0006540	.0006811
	4	OH	-19.7527	.0154000	.0152673
	5	H2O	-36.7180	3.9718994	3.971582
	6	CO	-30.1221	.0815971	.0819268
	7	CO2	-49.5104	2.9184028	2.9180737
	8	N2	-23.0912	19.9866573	19.986533
	9	NO	-20.5868	.0266857	.0269337
	10	O2	-24.9310	.0335845	.0338415
	11	O	-4.7912	.0004428	.000459

λ Total number of moles = 27.0553973 (1)
λ Total number of moles = 27.057600 (2)
Summation of molar fractions at optimum = 1.000001 (2)

Notes:
(1) = Dinkel and Lakshmanan, 1977
(2) = This thesis
N.A. = Not available

PROBLEM 6 : CLAUS FURNACE 1 Bonsu, 1981

Species = 8   Elements = 4   Phases = 1

G/RT = N.A.   (1)   Temperature = 800K
G/RT = 110.48998 (2)   Pressure = 1 atm.

i	Elements	Bi	Zi (1)	Zi (2)
1	S	0.3	N.A.	-7.01247
2	O	1.0	N.A.	-36.5171
3	H	2.0	N.A.	-12.6496
4	N	3.760	N.A.	-12.3856

k	j	Species	Cj	Xj (1)	Xj (2)
1	2	SO2	-76.38437	.02568	.02565
	3	H2S	-29.34294	.05136	.05139
	4	H2O	-60.55412	.28304	.28304
	5	S2	-9.51304	0.01098	0.01098
	6	S4	-19.09567	0.00013	0.00013
	7	S6	-33.14049	.00013	.00013
	8	S8	-43.41504	.00000	.00000
	9	N2	-24.30711	.62868	.62869

λ Total number of moles at equilibrium = N.A. (1)
λ Total number of moles at equilibrium = 2.99003 (2)
Summation of molar fractions at optimum = 1.00001 (1)
Summation of molar fractions at optimum = 1.000001 (2)

Notes:
(1) = Bonsu, 1981
(2) = This thesis.
N.A. = Not available.
The reference gives the composition in molar fractions.

<u>PROBLEM 6B</u> :<u>CLAUS FURNACE 1</u> Mc.Gregor, 1978

Species = 8 Elements = 4 Phases = 1

G/RT = N.A. (1)
G/RT = 230.12109 (2)

Temperature = 800K
Pressure = 1 atm.

i	Elements	Bi	Zi (1)	Zi (2)
1	S	2.0	N.A.	-6.28329
2	O	2.0	N.A.	-36.6006
3	H	4.0	N.A.	-12.7332
4	N	3.76	N.A.	-12.4229

k	j	Species	Cj	Xj (1)	Xj (2)
1	2	SO2	-76.38437	.0455	.04505
	3	H2S	-29.34294	.0908	.09009
	4	H2O	-60.55412	.2192	.21922
	5	S2	-9.51304	.0483	.04783
	6	H2	-16.96621	.0002	.00020
	7	S6	-32.48436	.0112	.01130
	8	S8	-43.75302	.0011	.00120
	9	N2	-24.30711	.5833	.58311

λ Total number of moles at equilibrium = N.A. (1)
λ Total number of moles at equilibrium = 6.44320 (2)
Summation of molar fractions at equilibrium = 1.0001 (1)
Summation of molar fractions at equilibrium = 1.000001 (2)

Notes:
(1) McGregor, 1978
(2) This thesis.
The reference gives the composition in molar fractions.

PROBLEM 7 : CLAUS FURNACE 2 Bennett,1979

Species =24	Elements = 4	Phases = 1
G/RT = N.A. (1)		Temperature = 800K
G/RT = 230.13785		Pressure = 1 atm.)

i	Elements	Bi	Zi (1)	Zi (2)
1	S	2.0	N.A.	-6.28617
2	O	4.0	N.A.	-36.6064
3	H	4.0	N.A.	-12.7318
4	N	7.52	N.A.	-12.4236

k	j	Species	Cj	Xj (1)	Xj (2)
1	2	SO2	-76.38437		.4440 E-1
	3	H2S	-29.34294		.9010 E-1
	4	H2O	-60.55412		.2197 E 0
	5	S2	-9.51304	N.A.	.4692 E-1
	6	S4	21.72258		.4424 E-20
	7	S6	-32.48436		.5339 E-2
	8	S8	-43.75302		.1450 E-2
	9	N2	-24.30711		.5827 E 0
	10	NH3	-31.83038		.6922 E-8
	11	S	20.62205		.2060 E-11
	12	SH	-2.09771		.4484 E-7
	13	H2	-16.96626		.2040 E-3
	14	H	18.08661		.4128 E-13
	15	SO	-27.24001		.1593 E-6
	16	HO	-17.51132		.1506 E-13
	17	SO3	-92.84698		.7925 E-10
	18	SN	11.90983		.5036 E-13
	19	S2O	-42.68584		.1514 E-2
	20	NO	-13.05837		.2386 E-15
	21	S3	-13.37118		.4139 E-2
	22	S5	-24.89094		.1445 E-2
	23	S7	-37.89094		.2186 E-2
	24	O	-17.17544		.4394 E-23
	25	O2	-25.98145		.1983 E-22

λ Total number of moles at equilibrium = 6.4500 (1) (3)
λ Total number of moles at equilibrium = 6.45207(2)
Summation of molar fractions at equilibrium = N.A. (1)
summation of molar fractions at equilibrium=1.0000001 (2)

PROBLEM 7 : CLAUS FURNACE 2 Continued

Notes:
(1) = Bennett, 1979
(2) = This thesis.
(3) = Calculated from Table 6-1 , p.141 ,Bennett,1979

Moles of product formed at 800 K from 100 moles of SH2
238 moles of air :

Species	Moles of product (1)	Moles of product (2)
H2	.06	.010
H2S	29.16	29.063
SO2	14.54	14.323
N2	187.91	187.976
H20	70.46	70.864
Sj	19.52	19.821
Others	.85	.545
Total	322.50	322.602

Notes :
(1) Table 6-1, p. 141 , Bennett, 1979
(2) This thesis.