10/-28-77
LA-NUREG-6500

780
NRC-8

MASTER

QUIL: A Chemical Equilibrium Code

Issued: February 1977

los alamos
scientific laboratory
of the University of California
LOS ALAMOS, NEW MEXICO 87545

Non Affirmative Action/Equal Opportunity Employer

UNITED STATES
ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
CONTRACT W-7405-ENG. 36

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

This work was performed under the auspices of the US Nuclear Regulatory Commission's Office of Nuclear Regulatory Research, Division of Reactor Safety Research.

Printed in the United States of America Available from
National Technical Information Service
U S Department of Commerce
5285 Port Royal Road
Springfield, VA 22161
Price Printed Copy \$5 00 Microfiche \$3 00

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Nuclear Regulatory Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

QUIL: A Chemical Equilibrium Code

by

J. L. Lunsford

Manuscript completed: August 1976
Issued: February 1977

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

Prepared for the US Nuclear Regulatory Commission
Office of Nuclear Regulatory Research

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

CONTENTS

ABSTRACT	1
I. INTRODUCTION	1
II. QUIL	4
III. THE SYSTEM SOLVER	8
IV. INPUT/OUTPUT	12
V. FENG	14
VI. SURF	15
VII. PROGRAM QUIL	19
VIII. SUBROUTINE QPUT	21
IX. SUBROUTINE QFIT	24
X. SUBROUTINE SSQS(QERR)	24
XI. SUBROUTINE APLT	25
XII. SUBROUTINE RPLT	25
XIII. PROGRAM FENG	26
XIV. SUBROUTINE FPUT	26
XV. SUBROUTINE FFIT	28
XVI. SUBROUTINE FPLT	29
XVII. PROGRAM SURF	29
XVIII. SUBROUTINE SPUT	30
XIX. SUBROUTINE SFIT	34
XX. SUBROUTINE SSQS(SERR)	35
XXI. SUBROUTINE SPLT	36

XXII. SUBROUTINE PAGE	36
XXIII. SUBROUTINE CRAC	37
XXIV. SUBROUTINE SYST	39
XXV. SUBROUTINE DSER	39
XXVI. SUBROUTINE TRIP	40
XXVII. SUBROUTINE PARA	41
XXVIII. PROGRAM LREF	42
APPENDIX A—SAMPLE SESSION WITH QUIL	42
APPENDIX B—SAMPLE SESSION WITH FENG	57
APPENDIX C—SAMPLE SESSION WITH SURF	68
REFERENCES	80

QUIL: A CHEMICAL EQUILIBRIUM CODE

by

J. L. Lunsford

ABSTRACT

A chemical equilibrium code QUIL is described, along with two support codes FENG and SURF. QUIL is designed to allow calculations on a wide range of chemical environments, which may include surface phases. QUIL depends upon an energy-data library called ELIB. This library is maintained by FENG and SURF. FENG enters into the library all reactions having standard free energies of reaction that are independent of concentration. SURF enters all surface reactions into ELIB.

All three codes are interactive codes written to be used from a remote terminal, with paging control provided. Plotted output is also available.

I. INTRODUCTION

QUIL is a code to solve complex equilibrium distributions in chemical environments. The label QUIL is an acronym taken from the word "equilibrium." As the name implies, the code makes calculations based upon equilibrium thermodynamics. QUIL was written specifically to calculate distributions associated with complex equilibria involving fission products in the primary coolant loop of the high-temperature gas-cooled reactor. In particular, the code is able to treat surface phases.

In execution, QUIL relies on a data file containing thermodynamic information. This energy library is called ELIB and is maintained by two support codes, FENG and SURF. The labels FENG and SURF are acronyms taken from the words "free energy" and "surface." Figure 1 illustrates the relationship between the three codes and the energy library. Embedded in FENG and SURF are parametric models by which to represent experimental data. FENG treats equilibria that have standard free energies of reaction which are functions of temperature independent of composition. SURF treats surface phases. In operation, equilibrium data are loaded into either FENG or SURF, and the code fits the model to the data. The parameters associated with the fit, the data, and any other pertinent information are then loaded into ELIB. All three codes use the same format and QUIL is thus able to avail itself of the data in the energy library.

To systematize the use of QUIL, FENG, and SURF, certain labeling conventions are adopted. First, all program names have four letters. Second, the main program in any code has the same name as the code. For example, the main program in QUIL has the program name QUIL. Third, the file name of a routine is the program name followed by the three characters FIL. Thus, the file name of the main program QUIL is QUILFIL. Fourth, the assembled main program with its

Fig. 1.
Relationship between the support codes FENG and SURF, the energy library ELIB, and the chemical equilibrium code QUIL.

associated subroutines has a file name which is the program name of the main program followed by the three characters COD. For example, the assembled code QUIL has a file name QUILCOD.

One exception exists to the rule for labeling files from program names. The direct-search routines are intended to be applicable as a library package. These routines call a subroutine SSQS to calculate phi, the sum of the squares of the errors for the problem. Therefore, the program names of the subroutines in the various codes that return phi must be identical, whereas the file names must be distinct. The subroutines to compute phi for QUIL and SURF have the file names QERRFIL and SERRFIL. (Because FENG is a linear fit, no call is made to the non-linear direct-search package.)

The direct-search package in QUIL and SURF manages a direct search on the sum of the squares of the errors in the code. The search is conducted as a function of two search parameters r and t. These parameters have labels in this report that are different from those used in the FORTRAN code. The FORTRAN label for the r parameter is PP, and the FORTRAN label for the t parameter is EE.

All codes are put into execution with a procedure file. The file name of the procedure file is the program name of the code it executes. Thus, QUILCOD is put into execution by a procedure file with a file name QUIL. Normally, the procedure file operates from the binary version of the code it executes. The binary version of a code is labeled by adding the three characters BIN to the program name. For example, the compiled version of QUIL is called QUILBIN, and the procedure file QUIL puts the code QUILCOD into execution with the binary file QUILBIN.

If the procedure file operates from the FORTRAN listings, it is identified by appending the three-letter label SUB to the file name. For example, the procedure file QUILSUB operates from the FORTRAN listings. First, the procedure file assembles QUILCOD from the subroutines, and QUILCOD is stored. Then the procedure file compiles QUILBIN, which is also stored. During the compilation, the FORTRAN listing is written to the file QUILFOR. Finally, the procedure file goes into execution with the binary version QUILBIN. From the foregoing, it is obvious that the procedure file XXXXSUB must be executed before the procedure file XXXX, because XXXX-BIN is loaded by XXXXSUB. In general, XXXXSUB must be executed after any change in the FORTRAN listings that make up the code XXXX. Once XXXXSUB has been executed, subsequent runs should be made with the procedure file XXXX, because compile times on a CDC-6600 range from 40 to 65 s for the three codes.

As mentioned above, all codes are put into execution with procedure files. The codes FENG and SURF alter the library in the normal course of operation. Consequently, the procedure files that execute FENG and SURF restore the updated library as ELIBFIL only if no errors occur in execution. As an additional safety feature, a backup copy of ELIB is written to a permanent file ELIBSAF at the start of execution.

The plotting routines rely on a commercial software package that was modified for use in QUIL, FENG, and SURF.* The entire package numbers some 22 routines. Of these, only five were modified. These modified routines are included in the report without description. The plotter software is incorporated in the binary file ZETABIN. The procedure files copy this file to the compiled version of the code before execution.

Codes QUIL, FENG, and SURF are written to output printed results to a terminal. Some provision is therefore needed for page control. A subroutine PAGE is incorporated into the codes for this purpose. As a result, those sections of the codes that provide for printed output reflect the more complicated coding structure. Two utility codes, LIST and PRNT, which can be used with PAGE to list from a terminal are also included. These codes are put into execution by a procedure file LIST.

*Zeta Research, 1043 Stuart St., Lafayette, CA 94549.

All interactive sessions must be conducted in the full ASCII convention, i.e., the terminal must be able to distinguish upper- and lower-case characters. Responses to prompts must be appropriate to the question, although the codes are so written that, where possible, either upper- or lower-case responses are correctly interpreted. For example, a correct response to a question might be indicated as YES(yes) or NO(no) when either an upper- or lower-case response will suffice.

The energy library ELIB has two sections, a reference section and a bibliography section. The reference section contains all numerical data and the bibliography section contains the reference sources. The two sections are separated by a line containing an asterisk in the first column. To begin a new library, one should start with a file containing a single line with an asterisk in column 1.

The reference section of the library contains the reactions and all associated numerical data. The reactions are written in the full ASCII convention. Each reaction is loaded beginning in column 1. No other data in the reference section may be entered into the first column. In the bibliography section, each entry follows a line with a # mark in column 1 and a number in columns 2-10. This number is the entry number of the reference into the bibliography section.

QUIL is presently dimensioned to accept as many as 20 elements, 40 species, 20 phases, and 20 reactions.

Typical sessions at the terminal are included for illustrative purposes. Interactive sessions with the codes QUIL, FENG, and SURF are found in Appendixes A, B, and C. All routines needed to execute QUIL, FENG, and SURF are included on the microfiche in the pocket inside the back cover of this report.

QUIL has recently been used to generate material for a report concerning the distribution of fission-product cesium in the gas-cooled reactor, and the reader interested in typical QUIL capabilities should refer to that report.¹

II. QUIL

Complex equilibrium codes compute the distribution of species present in chemical environments. While there are many numerical techniques by which to carry out such calculations, most approaches fall into one of two categories: either the code represents a free energy minimization algorithm or a nonlinear system solver. In either case, the goal is to specify the state of the system at equilibrium. In this sense, the calculation represents an implementation of equilibrium thermodynamics.

QUIL is an example of a nonlinear system solver. The complex equilibrium problem is defined by stating a temperature, a pressure, those chemical elements whose total amounts are to be conserved (the gross stoichiometry), the species assumed to be present at equilibrium, the reactions by which to link these species, and the free energies or equilibrium constants of these reactions. From the stoichiometry, we get a set of mass balance equations, and from reactions we get a set of equilibrium relations. The temperature and the pressure specify the equilibrium conditions. The necessary thermodynamic data can then be obtained from standard tables or by other means, and the set of mass balance and equilibrium equations can be solved by whatever method is selected.

In QUIL, we take A,B,...,L,M,... as the species present that are linked by a general reaction of the form

where the values $a, b, \dots, l, m, \dots$ are the species reaction coefficients and ΔG° is the accompanying free energy change in the standard state. If we have M_K moles of a species K in a phase J containing a total of M_J moles of all species at a total pressure P , we may take the activity A_K of any species present as

$$A_K = \frac{M_K}{M_J} p^{\delta_j} \quad (2)$$

where δ_j is a delta function

$$\delta_j = \begin{cases} 0, & j \neq \text{gas}, \\ 1, & j = \text{gas}. \end{cases} \quad (3)$$

From the definition of the equilibrium constant, we have

$$\frac{A_L^l A_M^m \dots}{A_A^a A_B^b \dots} = K^\circ, \quad (4)$$

where the A 's are the activities of the set of species present. The equilibrium constant K° is related to the free energy change ΔG° by an expression of the form

$$\Delta G^\circ = -RT \ln(K^\circ), \quad (5)$$

where R is the gas constant and T is the absolute temperature.

Accuracy in large system solvers is highly variable, depending upon many factors such as system size and conditioning. Equilibrium constants as defined in Eq. (4) have values that typically range from 10^{+60} to 10^{-160} . If one chooses to define error components in the product form of Eq. (4), the wide variation in the values of the equilibrium constants immediately suggests that some of the error conditions will be smaller than machine accuracy. When this happens, the equilibrium condition as written does not constrain the system of equations as solved. Accordingly, we elected to work with a logarithmic form of Eq. (4). If one denotes the coefficient of the species K in equation N by S_{NK} , then the definition of the equilibrium constant can be written in logarithmic form as

$$\sum_K S_{NK} \ln(A_K) = \ln(K_N^\circ), \quad (6)$$

where the coefficients on the left-hand side of Eq. (1) are taken as negative. Accordingly, we take the set

$$(\text{ERR})_N = \sum_K S_{NK} \ln(A_K) - \ln(K_N^\circ) \quad (7)$$

as N error conditions which the system solver must drive to zero.

QUIL, which employs both the Gauss-Newton and the Newton-Raphson methods, requires the Taylor's series expansion through the linear term of the various error components in the problem. The independent variables are taken as the moles of species present in the chemical environment. We therefore require the partial derivatives of these various error components with respect to M_L ,

the moles of species L which are found in M_I , the moles of phase I. The partial derivative of an equilibrium error component becomes

$$\frac{\partial (ERR)_N}{\partial M_L} = \frac{S_{NL}}{M_L} - \sum_K \frac{S_{NK}}{M_{I(K)}} . \quad (8)$$

where the $I(K)$ indicates that the summation extends over all species K common to the phase I. Activities are frequently as small as 10^{-50} . For species that are present in such small amounts, the leading term in Eq. (8) dominates since it is inversely proportional to M_L . Thus, these partials which are the components of the normal matrix tend to very large numbers and the value of the determinant tends to infinity. This problem can be resolved by the change of variable

$$Q_K = \ln(M_K) . \quad (9)$$

Since

$$\frac{\partial Q_K}{\partial M_K} = \frac{1}{M_K} \quad (10)$$

and

$$\frac{\partial (ERR)_N}{\partial Q_K} = \frac{\partial M_K}{\partial Q_K} \frac{\partial (ERR)_N}{\partial M_K} , \quad (11)$$

we have

$$\frac{\partial (ERR)_N}{\partial Q_K} = M_K \frac{\partial (ERR)_N}{\partial M_K} \quad (12)$$

or

$$\frac{\partial (ERR)_N}{\partial Q_L} = S_{NL} - \sum_K S_{NK} \frac{M_L}{M_{I(K)}} . \quad (13)$$

These partials are thus seen to be of the order of S_{NL} , since the ratios (M_L/M_I) range from zero to one. The matrix components arising from the products of these partials are therefore of the order of unity.

If the various species are composed of M elements, there are M mass balances which can be written as conservation conditions. First denote the number of moles of an element M in one mole of a species K as the element reaction coefficient E_{MK} . If the number of moles of an element present in the system is M_M , then the M mass balances may be written

$$\sum_K E_{MK} M_K = M_M . \quad (14)$$

From these we can write the set

$$(\text{ERR})_M = \sum_K E_{MK} \frac{M_K}{M_M} - 1 \quad (15)$$

as M error conditions which the system solver must also drive to zero.

The partial derivative of a stoichiometric error component becomes

$$\frac{\partial (\text{ERR})_M}{\partial Q_L} = M_L \frac{\partial (\text{ERR})_M}{\partial M_L} \quad (16)$$

or

$$\frac{\partial (\text{ERR})_M}{\partial Q_L} = M_L E_{ML} \quad (17)$$

When species are present in small amounts, these partials, which are the components of the normal matrix, tend to very small numbers and the value of the determinant tends to zero. This problem can be resolved by dividing the stoichiometry relations of Eq. (15) by M_M . With this change, we have

$$(\text{ERR})_M = \sum_K E_{MK} \frac{M_K}{M_M} - 1 \quad (18)$$

and

$$\frac{\partial (\text{ERR})_M}{\partial Q_L} = E_{ML} \frac{M_L}{M_M} \quad (19)$$

These partials are of the order of E_{ML} since the ratios (M_L/M_M) range from zero to one. The matrix components arising from the product of these partials are therefore of the order of unity.

In summary, we compute the Taylor's series expansion for N error conditions of the form

$$(\text{ERR})_N = \sum_K S_{NK} \ln(A_K) - \ln(K_N^O) \quad (20)$$

and M error conditions of the form

$$(\text{ERR})_M = \sum_K E_{MK} \frac{M_K}{M_M} - 1 \quad (21)$$

to solve for the K values of Q_K , where

$$Q_K = \ln(M_K) \quad (22)$$

and

$$M + N = K. \quad (23)$$

The overall system, written thus, minimizes the possibility of ill-conditioning, since all matrix components are of the order of unity.

The activities of the species present in a typical equilibrium distribution may vary by 50 to 60 orders of magnitude. Species that are present in trivial quantities may make up the total amount from a particular element. Furthermore, species that are insignificant over a limited range of a calculation may become important in some other part of the range. If one includes multiple phases in the calculation, approximate or stepwise solution techniques obviously become difficult to apply. QUIL, operating as a single-pass, nonapproximate system solver, has extracted solutions with well-behaved minima when concentrations differed by 160 orders of magnitude.

III. THE SYSTEM SOLVER

It was desirable that QUIL should be able to handle a wide variety of problems and yet be efficient with regard to running time. As the more powerful system solvers and search techniques usually require more time in execution, QUIL is written in levels so that the simplest method applicable is always the one in use. At each stage of the calculation, various tests are made for approach to convergence, ill-conditioning, singularity, etc., and appropriate changes are made in the solver/search strategy. In essence, QUIL changes techniques under program control to meet the conditions existing at any point during execution. This feature allows the code to optimize itself continually against the problem during execution and increases its applicability to a wider range of problems.

The complete specification of the search strategy is too extensive to be covered here. However, the general features of the system solver and the direct-search philosophy are detailed below. QUIL employs both the Gauss-Newton method and the Newton-Raphson method. The first technique emphasizes speed, whereas the second is a more powerful system solver. During each iteration, one of the two methods is used exclusively. Therefore, at the start of each iteration, a decision is made as to which technique to employ.

The Taylor's series expansions of the K error components have been evaluated in the previous section. With a change in notation, the form of the linearized expansion of an error component E_n becomes

$$E_n = E_{n0} + \sum_{k=1}^K \frac{\partial E_n}{\partial p_k} q_k ; n = 1, \dots, K, \quad (24)$$

where q_k is the incremental change in the parameter p_k . Let the matrix component a_{nk} be written

$$a_{nk} = \frac{\partial E_n}{\partial p_k} . \quad (25)$$

With the vectors e , p , and q defined as

$$e = (E_{10}, E_{20}, \dots, E_{K0})^T, \quad (26)$$

$$p = (p_1, p_2, \dots, p_K)^T, \quad (27)$$

and

$$q = (q_1, q_2, \dots, q_K)^T, \quad (28)$$

we have the Taylor's series expansion in matrix form as

$$Aq = -e, \quad (29)$$

where q is the incremental change in p for $E_n = 0.0$.

QUIL first attempts a solution by the Gauss-Newton method. Multiplying Eq. (29) by A^T , we have

$$(A^T A)q = Bq = -A^T e, \quad (30)$$

where B is the normal matrix for the system. Multiplying Eq. (30) by the inverse of the normal matrix, we have

$$q = -B^{-1} A^T e, \quad (31)$$

where q is now the vector correction to p .

Equation (31) is solved in subroutine SYST which calls a standard library routine that employs partial pivoting and double precision accumulation of inner products. This standard library routine has the advantage of being fast. However, as Eq. (31) indicates, the Gauss-Newton method requires considerable computational manipulation of the original Taylor's series expansions. When the original system of equations is pathological, the formation of the inverse $(A^T A)^{-1}$ increases ill-conditioning, and the Gauss-Newton method will give poor results. To test for this eventuality, the routine is entered twice. On the first pass, the determinant of the system is computed and Eq. (30) is normalized. On the second pass, the value of the determinant should be 1.0. If the determinant is 1.0 to preset limits, the Gauss-Newton method is used for all further calculations required during that iteration.

If the second pass through the standard library routine reveals that the determinant is no longer 1.0 to desired accuracy, a flag is set to indicate that the input system is ill-conditioned, and QUIL resorts to the Newton-Raphson method to solve Eq. (29). Since the Newton-Raphson method works with the original Taylor's series expansion [Eq. (29)], deterioration of conditioning is avoided. This is an important feature of the technique. To solve the original system, a second slower but more powerful library routine is then called to extract the solution. This more generalized system solver employs the method of minimum least squares as proposed by Osborne.² This generalized routine can extract solutions in the least-squares sense from systems of equations that are under-determined, square, or over-determined. For the remainder of the iteration, the Newton-Raphson method is employed for all calculations.

Direct-search capabilities in QUIL are derived from two parameters which may be varied to produce local minima in the computed value of the sum of the squares of the errors. The first is the Levenberg-Marquardt parameter.^{3,4} To illustrate how this parameter is used, we rewrite a modified version of Eq. (29) in partitioned form as

$$\begin{bmatrix} A \\ r^{1/2} I \end{bmatrix} \begin{Bmatrix} q \\ o \end{Bmatrix} = \begin{Bmatrix} -e \\ o \end{Bmatrix}, \quad (32)$$

where o is a null vector of dimension K , I is the unit matrix of dimension $K \times K$, and r is the Levenberg-Marquardt parameter. In light of Eq. (32), Eq. (30) becomes

$$(A^T A + rI)q = B'q = -A^T e, \quad (33)$$

which redefines the normal matrix for the system.

QUIL is written to minimize a function ϕ which is the sum of the squares of the error components E_k as defined in Sec. II. Thus

$$\phi = \sum_{k=1}^K E_k^2. \quad (34)$$

The negative gradient g is the slope of ϕ in parameter space. Thus

$$g_j = -\frac{\partial \phi}{\partial p_j} = -2 \sum_{k=1}^K E_k \frac{\partial E_k}{\partial p_j}; \quad j = 1, \dots, K, \quad (35)$$

or, in vector form,

$$g = -2A^T e. \quad (36)$$

Combining Eqs. (33) and (36), we have

$$(A^T A + r I) q = \frac{g}{2}. \quad (37)$$

From Eq. (37), we can see the effect of r upon the solution vector q . When r tends to zero, q tends to the unperturbed Gauss-Newton solution. When r tends to infinity, q tends to $g/(2r)$. The advantage of the r parameter is that it can cause a continuous rotation of the solution vector from the negative gradient direction to the Gauss-Newton direction. An analogous argument holds for the Newton-Raphson method.

The disadvantage is the effect the parameter has on the norm of the solution vector. To compensate for this second effect, we introduce a second parameter t which is simply a factor by which to scale the solution vector. Thus, as a result of the i^{th} iteration, we replace the current value of the parameter vector, p_i , with an updated version

$$p_{i+1} = p_i + tq(r) \quad (38)$$

as a starting point for the next iteration. Both t and r are subjected to direct-search techniques.

Figure 2 indicates schematically a hypothetical search that might take place during a typical iteration. When the r parameter is employed, considerable computing time is expended since the system solver must be entered for each calculation. To minimize the use of the r method, the calculation is initiated as a one-dimensional search in the t direction with $r = 0$. The search proceeds in two phases. In the first phase, a set of three equidistant points is moved along the t axis in what is here called a tripod search. The location and spacing of these three points are varied until a minimum t_1 is located with a satisfactory standard deviation of the sum of the squares of the errors at the three points in question. The second phase of the search, called here a parabolic search, is conducted by fitting a parabola to these three points, computing a new predicted minimum, and selecting a new set of points by which to continue the algorithm. In this selection process, the lowest point and its nearest neighbor on each side are always selected as the new set of values with which to continue the search. In this manner, the iterative process is guaranteed to be stable. When this algorithm fails to find a new minimum smaller than the old minimum, the parabolic search is abandoned. The tripod search results in a set of three values of

Fig. 2.

A hypothetical search as conducted by the subroutine DSER. The search proceeds along the t axis to the minimum t_1 and then into the r - t plane along semicircles with progressively larger radii until the minimum at R_3 is located.

t clustered about the final minimum, from which we calculate a radius which is half the span of these three values. The parabolic search results in an improved estimate of that minimum.

With these values of the minimum and a radius, the two-dimensional search is entered, called here a semicircular search. The second search variable is the angle of the radius vector from the minimum t_1 . A search then proceeds in two phases as described above. First a tripod search locates a minimum with an acceptable standard deviation. Then a parabolic search improves the estimate of that minimum. The radius is then doubled, and the same procedure is carried out at the new radius. When the search conducted at the larger radius reveals a new minimum which is larger than the old minimum, the semicircular search is abandoned. Values of t_2 and r_2 at the minimum are stored as adjuncts to the final solution.

When large gains are being made by the one-dimensional t search, the semicircular search rarely reveals lower minima. Therefore, simple problems tend to operate without entering into the t - r plane for more than one pass, and computing time is conserved. When only slight gains are effected by the one-dimensional search, the semicircular search almost always reveals a minimum that is lower than that found at t_1 and the search travels out into the t - r plane. Thus the search strategy tends to adjust itself automatically and the use of computing time is optimized.

When the correction vector becomes smaller than a preset limit, the direct search is by-passed, and the direct jump is taken with $r = 0.0$ and $t = 1.0$. Convergence is assumed when successive iterations do not reduce the sum of the squares of the errors after a direct jump.

Since the initial stoichiometries may differ by many orders of magnitude, a good starting approximation to the solution vector is important to minimize execution time. An initial estimate is achieved in the following manner. First, one selects that element E which is present in the smallest quantity M_E . If there are L species present that contain that element, then the L species are all assigned a starting value of M_E/L moles. The program cycles through the elements present from the least abundant to the most abundant in this fashion. In this way, a starting solution is obtained that is within several orders of magnitude of the final answer for all but a few species. This is a trivial computational step, but it saves a great deal of computing time.

If only one phase is present when the code goes into execution, the code first enters a preprocessor mode. In this mode the initial approximate solution is used to calculate an improved distribution by assuming the equilibrium constants are all one and the stoichiometry will dictate the distribution. When QUIL converges in the preprocessor mode, the code resets the equilibrium constants to their proper values. This last approximation to the solution then becomes the starting point for calculating the final distribution.

IV. INPUT/OUTPUT

It was intended that QUIL be fast and flexible so that questions involving complex equilibria could be simply posed and quickly answered. The code could thus be used as an analytical tool in the study of the chemistry of the primary coolant loop of the high-temperature gas-cooled reactor.

To this end, several convenience features have been incorporated into QUIL. To begin with, QUIL is an interactive code. It is put into execution without input data. Once in execution, the code proceeds to ask the user for certain information through the medium of a keyboard terminal. This means that the input is format-free and that input branching takes place under program control.

As another convenience feature, QUIL terminates in a front-end preprocessor that accepts as input a chemical statement of the problem. By this is meant the equilibrium equations in conventional form. Section XXIII indicates input that QUIL would interpret as a problem statement. The code recognizes the conventional meaning of the mnemonic relations. In addition, QUIL will

accept relations that are tagged with the standard free energy of reaction G or the logarithm of the equilibrium constant K . The numerical expression used to tag the mnemonic expression must be of the form given by theory.⁵ This form is a power series in the temperature T to which is added a term in $\ln(T)$. In particular, a term $T \ln(T)$ may be added to the expression for the free energy of reaction and a term in $\ln(T)$ may be added to the expression for the logarithm of the equilibrium constant. For reactions where no equilibrium data are input, QUIL goes to the library.

The system solver in QUIL needs a numerical array. The input indicated is a mnemonic array. One implies the other, and as a further convenience to the user, QUIL deduces the numerical array implied by the mnemonic array. To do this, QUIL first breaks down the equations as written to determine the elements present, the species present, and the element and species coefficient matrices. It then proceeds to write the equilibrium relations as N Taylor's series expansions of error conditions that must be driven to zero in order to solve the problem.

At this point, the problem is still not soluble. There are more unknowns than conditions because the stoichiometry relations have been left out. However, these relations are not unique since they are implied by the set of equilibrium relations. Again, as a convenience to the user, QUIL writes the mass balance conditions as M Taylor's series expansions of error conditions that must be driven to zero to solve the problem. These $M + N$ error conditions are written in terms of the moles of the K species present in the starting stoichiometry where $K = M + N$.

Other features have been incorporated into QUIL to increase its convenience and utility. Of interest are instances of mixed equilibria, wherein certain reactions in a system are assumed to be in equilibrium while others are not. For example, one might wish to partition equilibrium in a given system by assuming gas-gas equilibrium in the absence of gas-solid equilibrium. The code will allow such conditions of partitioned equilibria to be imposed, and will calculate the potential to occur of reactions which are assumed to be out of equilibrium. These "reaction potentials" are then plotted over the range of the calculation.

QUIL will also allow the reactions in the library to be loaded as sums and differences. Thus reaction potentials can be computed for all possible reactions in the system. As an example, consider the two reactions

and

They could be solved under equilibrium assumptions to give the equilibrium distribution of species present. In this case there would be $\text{CO}_2\langle g \rangle$, $\text{CO}\langle g \rangle$, $\text{O}_2\langle g \rangle$, and $\text{C}\langle s \rangle$. The presence of solid carbon affects the resultant distribution significantly, and is equivalent to assuming that gas-solid equilibrium prevails. In the event that this assumption seems unwarranted, it is possible to solve for the following distribution instead. If one takes the difference of the two reactions presented above, one obtains

as a single reaction. The resultant distribution of species, when solved as above, is exactly that which would prevail if the system had no free carbon in it. In effect, the system has been solved with the assumption that gas-gas equilibrium prevails and that gas-solid does not.

Having excluded $C_{<s>}$ from the calculation, it is now of interest to know if the environment as calculated is predisposed to precipitate carbon or to react with it. Consider the definition of the equilibrium constant for the first reaction written above in Eq. (39)

$$K = \frac{P_{O_2}}{P_{CO_2}} . \quad (42)$$

Since the reaction was not included in the system as solved in the second instance [Eq. (41)], the equality as written in Eq. (42) is not satisfied. If one defines

$$K' = \frac{P_{O_2}}{P_{CO_2}} , \quad (43)$$

where K' is computed from the existing distribution solved in the absence of $C_{<s>}$, then the ratio

$$R = \frac{K}{K'} \quad (44)$$

may be taken as a reaction constant. It is a measure of the extent to which the CO_2 reaction of Eq. (39) is out of balance in the reduced system of Eq. (41) and is thus a measure of the tendency or "potential" of the system to precipitate carbon from the gaseous environment assumed to be in equilibrium. The logarithm of the ratio of these two constants is then defined as a reaction potential. A positive reaction potential indicates a tendency for the reaction in question to run to the right as written.

V. FENG

A large body of chemical reactions is characterized by having standard free energies of reaction that are functions of temperature independent of concentration. For such reactions, FENG fits a parametric model to the equilibrium data and loads the fit into ELIB. The model for the fitting process is a power series in the temperature T , with a term added in $T \cdot \log(T)$ which is given by theory.⁵ The result is an expression of the form

$$\Delta G = a_{-1} T \log(T) + \sum_{n=0}^{10} a_n T^n , \quad (45)$$

where ΔG is the standard free energy of reaction. The fit is initiated with all 12 parameters free in the fit. If there are fewer than 13 data points to be fitted, the initial number of free parameters is taken as one less than the number of data points. If the fitting process should fail for any reason, the parameter associated with the highest order term present in the series is removed by nulling, and the fit is repeated with one less parameter.

The fitting algorithm requires that all parameters in the final fit be statistically significant. In FENG, the test of significance is taken as the requirement that the estimate of the parameter exceed by a factor of 2 the estimate of the standard deviation of the parameter in the fit. If any of the parameters as fitted are insignificant, that parameter which is numerically the least significant is removed by nulling, and the fitting procedure is repeated until all parameters active in the fit are statistically significant.

The power series as fitted allows the calculation of the standard free energy of reaction for a specified temperature. From the free energy, we can calculate the equilibrium constant

$$\ln(K) = - \frac{\Delta G(T)}{RT}, \quad (46)$$

where R is the gas constant. With the equilibrium constant, it is possible to write an equilibrium relation of the form

$$K = \frac{A_L^1 A_M^{\dots}}{A_A^a A_B^b \dots}, \quad (47)$$

where the A 's are the activities of the species $A, B, \dots, L, M, \dots$ present in the chemical environment (see Sec. II).

A typical fit carried out by FENG to the thermodynamic data for the H_2O reaction is included in Appendix B.

VI. SURF

Chemical reactions involving surface phases are characterized by having standard free energies of reaction that are functions of temperature and composition. For such reactions, SURF fits a parametric model to the equilibrium data and loads the fit into ELIB. The model for the fitting process is nonphysical, and assumes that any function can be approximated by a series of straight-line segments with appropriate melding parameters to fair the segments into one another. The model is an expression of the form

$$P = P_o \prod_{m=1}^M Z_m^{S_m}, \quad (48)$$

where there are $(M+1)$ straight-line sections in a log-log plot of the data to be fitted. The asymptotic structure of the model is shown in Fig. 3 for a hypothetical situation where there are three breakpoints. Each function Z_m is effective in the region between breakpoint m and breakpoint $(m+1)$, where the breakpoints are numbered from right to left. S_m is the slope of the fitted function in a log-log plot of pressure vs coverage, also in the region between breakpoint m and breakpoint $(m+1)$. The term P_o is the limiting pressure at constant temperature, and is effective in the region to the right of the first breakpoint. This limiting pressure is taken to be the equilibrium pressure of the species in question above a condensed phase at unit activity, either liquid or solid.

The functions Z_m are expressions of the form

$$Z_m = \frac{C_{m+1} \left(1 + u_{m+1}^{F_{m+1}}\right)^{G_{m+1}}}{C_m \left(1 + u_m^{F_m}\right)^{G_m}} \quad (49)$$

Fig. 3.

A hypothetical breakpoint model for SURF with three breakpoints. The functions Z_m control the model in the region between breakpoint m and breakpoint $(m+1)$, where the breakpoints are numbered right to left. The limiting pressure P_o is the equilibrium pressure of the species in question above a condensed phase at unit activity.

with

$$u_m = \frac{C}{C_m}, \quad (50)$$

where C is the coverage, C_m is a breakpoint in the coverage, and

$$G_m = \frac{1}{F_m}. \quad (51)$$

The regions are numbered from right to left. For the leftmost region we have $m = M$, and

$$Z_M = \frac{u_M}{\left(1 + u_M\right)^{G_M}}. \quad (52)$$

The breakpoints C_m in the model are related by an expression of the form

$$\log(C_m) = \log(C_{m+1}) + \log(R_m), \quad (53)$$

or

$$\log(C_m) = \log(C_1) - \sum_{j=1}^{m-1} \log(R_j); \quad m = 1, \dots, M, \quad (54)$$

where C_1 is the concentration of the rightmost breakpoint in the model and

$$R_j = \frac{C_m}{C_{m+1}} \quad (55)$$

is the ratio of neighboring breakpoints. If there are M breakpoints in the model, Eq. (48) will be controlled by C_1 , $(M-1)$ values of R , M values of S , and M values of F . This gives a total of $3M$ parametric functions that must be calculated for the model. These functions are computed by expressions of the form

$$h(T) = \frac{a}{T} + b + c \cdot T. \quad (56)$$

The values of the functions are treated as logarithmic quantities. Therefore, except for C_1 , negative values of the breakpoint constants correspond to nonphysical situations. Consequently, the function $h(T)$ is embedded in a masking function $g(T)$, where

$$g(T) = 10^{-d \cdot \log \left(1 + 10^{+d \cdot h(T)} \right)}. \quad (57)$$

When $10^{+d \cdot h(T)}$ is large and positive, $g(T) = h(T)$. When $10^{+d \cdot h(T)}$ is large and negative, $g(T) = 0$. In the transition region, the relation is strongly dependent upon d . Consequently, d is taken as a free parameter in the fit. For C_1 , we set $g(T) = h(T)$, and the masking function is not used.

In summary, the model is determined by the values of C_1 , R , S , and F . For C_1 , we have an expression of the form

$$\log(C_1) = h(T). \quad (58)$$

For R , S , and F , we have expressions of the form

$$\log(R) = g(T), \quad (59)$$

$$\log(S) = g(T), \quad (60)$$

and

$$\log(F) = g(T), \quad (61)$$

where

$$g(T) = 10^{-d} \log(1 + 10^{10+d} h(T)) . \quad (62)$$

and

$$h(T) = \frac{a}{T} + b + cT. \quad (63)$$

Approximately 12 parameters are associated with each breakpoint. In practice, it is never desirable to free that many parameters in a fit. This redundancy is necessary, however, to insure that the parametric model has the flexibility to fit the varying situations that may arise.

The parametric model, as fitted, allows the calculation of the pressure of the adsorbate over the adsorbent as a function of the temperature and the coverage. The reaction indicated is of the form

$$Q[s,l] = Q <g>; \Delta G(T,C) . \quad (64)$$

Here the square brackets indicate a surface state, and the reaction is one by which the species Q goes from the adsorbed state on the condensed phase, solid or liquid $[s,l]$, to the gaseous state $<g>$.

To convert the parametric model of Eq. (48) into a form similar to Eq. (64), we begin with the reaction

$$Q <s,l> = Q <g>; \Delta G(T) . \quad (65)$$

Here the triangular brackets indicate a macroscopic state, and the reaction is one by which the species Q goes from the condensed state, solid or liquid $<s,l>$, to the gaseous state $<g>$. If the condensed phase is present at unit activity, we may write for the equilibrium constant for the reaction of Eq. (65)

$$K = P_Q, \quad (66)$$

where P_Q is the pressure of the species Q over the condensed phase. Consequently, we have

$$\ln(K) = \ln(P_Q) = - \frac{\Delta G(T)}{RT} . \quad (67)$$

Let us identify the pressure P_Q with the limiting pressure P_o of Eq. (48)

$$P_o = P_Q . \quad (68)$$

Also, let us define a pseudo-activity for the adsorbate to be

$$A_{[Q]} = \prod_{m=1}^M z_m^{S_m} . \quad (69)$$

We may then define the pseudo-reaction

$$Q[s,1] = Q < g > ; \Delta G(T) . \quad (70)$$

where

$$K = \frac{A < Q >}{A_{[Q]}} = \frac{P_Q}{\prod_{m=1}^M z_m^{S_m}} \quad (71)$$

and

$$\ln(K) = - \frac{\Delta G(T)}{RT} . \quad (72)$$

Notice that the standard free energy of reaction for the pseudo-reaction of Eq. (70) is no longer a function of the coverage.

Equation (70) is written in terms of a gaseous species that is monatomic. In the event that the gaseous species is polyatomic, Eq. (70) can be modified without loss of generality to

$$Q_n[s,1] = Q_n < g > ; \Delta G(T) , \quad (73)$$

where there are n atoms in the gaseous molecule.

A typical fit carried out by SURF to an iodine sorption isotherm is included in Appendix C.

VII. PROGRAM QUIL

The main program in QUIL executes all printed output. Input is from the keyboard, which is symbolic tape 1. Symbolic tape 2 contains the library. Running dumps, when activated, are to the terminal, which is symbolic tape 3. Printed output is to symbolic tape 4. Plotter output is to symbolic tape 5. A miscellaneous tape dump is maintained in the code as symbolic tape 11. This input/output capability is useful for dumping results of special runs.

QUIL is put into execution by the procedure file QUIL or QUILSUB. Both procedure files get symbolic tape 7 from a permfile called QUILDAT before QUIL goes into execution. During execution, all input for the problem is written to symbolic tape 8. After a normal exit from execution, the procedure file stores tape 8 as QUILDAT. The file QUILDAT may thus be regarded as the input data deck which is created during the interactive session with the user.

Restart capabilities are inherent in the back-up file tape 8. If a run must be repeated for any reason, the interactive session at the keyboard can be by-passed, and all current variables are updated from tape 7.

QUIL first reads input comments by which to label the run. It then exits to QPUT, where the first interactive session with the user is conducted. After this session, control returns to QUIL where a second interactive session is conducted with the user.

The code begins the second session by obtaining the temperature, pressure, and the stoichiometry for the run. One of these three inputs may be in the form of a "do loop." For example, temperature might be entered as (1500,300,-50), requiring that calculations be made every 50 degrees from 1500 K to 300 K. Also, it is possible to loop the stoichiometry with a logarithmic variation in one of the species. If the looping causes as many as four increments in the independent variable, plotting takes place automatically. Printing occurs only for those calculations specifically requested during the interactive session.

One or more elements frequently may be of interest in a particular calculation. The user is asked during the interactive session for special elements. If the user specifies one or more elements, the code will output additional information specific to those elements. In particular, additional plotting with only those elements of choice will be included.

After the second interactive session, all variables are dumped to the backup tape, which is symbolic tape 8. The code next sets up the temperature, pressure, and stoichiometry loops. Then it branches to QFIT, which manages the fitting procedure. Upon return, any printed output required is written to tape 4. If the loops are unsatisfied, control passes back to the reentry point in the code, the appropriate variable is stepped, and the calculation is repeated. If the loops are satisfied, the code branches to APLT where the distribution of species is plotted against the appropriate variable and then branches to RPLT which plots each reaction potential separately.

The prompts and responses for QUIL are included below. The period (.) is not typed as part of the response.

INPUT RUN COMMENTS.

1. Carriage return. Causes program jump to read input from symbolic tape 7.
2. Comment(s). Causes program to read identifying comment(s). (May not exceed two lines.)

TEMPERATURE(K)?

1. T. Causes the calculation to be carried out at T K.
2. T₁,T₂,T₃. Causes temperature to be looped from T₁ to T₂ in increments of T₃. (Only one loop may be active.)

PRESSURE(kPa)?

1. P. Causes the calculation to be carried out at P kPa.
2. P₁,P₂,P₃. Causes pressure to be looped from P₁ to P₂ in increments of P₃. (Only one loop may be active.)

MOLES [species]

1. Carriage return. Causes [species] moles to be set to zero.
2. m. Cause [species] moles to be set to m.

3. m_1, m_2, m_3 . Causes [species] moles to be looped from m_1 to m_2 in increments of m_3 . (Only one loop may be active.)
4. m_1, m_2, n . Causes [species] moles to be looped from m_1 to m_2 in n increments. (Only one loop may be active.)
5. $m_1, m_2, m_3, 1$. Causes [species] moles to be looped from m_1 to m_2 in increments of m_3 , where molar quantities input are logarithmic. (Only one loop may be active.)
6. $m_1, m_2, n, 1$. Causes [species] moles to be looped from m_1 to m_2 in n increments, where molar quantities input are logarithmic. (Only one loop may be active.)

INPUT SPECIFIC AREA FOR [species].

1. Carriage return. Causes specific area loaded with the library fit to be entered as the specific area for the adsorbent. (Units will be m^2/mol .)
2. n . Causes a new entry for the specific area of the adsorbent. (Units must be m^2/mol .)

ELEMENTS OF INTEREST?

1. Carriage return. No elements of special interest.
2. YES(yes). Causes program jump to input special elements.
3. NO(no). No elements of special interest.

M ELEMENTS

1	E_1
2	E_2
.	..
m	E_m

1. n . Causes element E_n to be tagged for special output.
2. Carriage return. Terminates tagging of special elements.

NUMBER OF OUTPUT SETS?

1. Carriage return. All sets are printed.
2. n . Causes n evenly spaced output sets to be printed. (n may not be less than 2.)

Appendix A gives a typical session at the terminal in which a problem is run.

VIII. SUBROUTINE QPUT

The routine QPUT conducts the first interactive session in the code QUIL. In this session, QPUT obtains the data necessary to write the matrices that are used to generate the nonlinear system solver. In essence, QPUT is the interface between the computer and the user. It converts the chemical information in mnemonic form to numerical arrays in matrix form.

QUIL can accept input in several forms. It identifies input from the keyboard in chemical mnemonic form that is tagged with free energy information. Alternatively, it can accept reactions from the library which are identified by their entry number. Reactions can be entered into the system that are the sum and difference of individual reactions residing in the library. Also, groups

of reactions are frequently of interest. QPUT is so written that any such grouping that has been predefined in a QPUT DATA declaration statement can be input with a single identifier. Format restrictions on the definitions of these collections are discussed in Sec. XXVIII. After QPUT has the list of equations that constitute the problem, it goes to the library for the free energy data.

QUIL can handle multiple phases; consequently, one must further specify the states of the various species. Therefore, the next operation in QPUT is to determine in which phases the various species are to be found. This is done in the interactive session in which QPUT prints each reaction in turn, and the user responds with the state specification. During this input, reaction potentials are flagged.

Sometimes the calculation must be restricted in the sense that certain types of equilibrium are not permitted. For example, one might wish to restrict gas-solid equilibrium in a system. Conventionally, this could be done implicitly by writing the reactions in such a way that solids do not appear. If, however, surface phases are present that are adsorbed on such solids, the solids cannot be removed. Instead, they are removed explicitly by tagging the elements that make up the particular species. The form of the tag is a /n/, where n may be any single-digit integer. This is referred to in the code as element linkage, since certain reactions are specifically linked to one another when the elements are so tagged. The next operation in QPUT is to input any element linkage indicated by the user.

At this point, the routine generates the system solver matrices from the reactions in storage. The compilation is carried out with the help of a subroutine CRAC (see Sec. XXIII). This routine processes each alphanumeric reaction in turn, and generates the numeric information implied by the equation. With this information, QPUT writes the system solver.

QUIL requires that certain variables be regenerated each time the temperature is incremented. The temperature loop does not involve QPUT. However, the variable arrays necessary to calculate these quantities are common to those already in QPUT. To conserve memory, the calculation is carried out in QPUT under an entry point labeled TPUT.

The prompts and responses for QPUT are included below. The period (.) is not typed as part of the response.

INPUT REACTIONS IN FULL ASCII.

1. He<g>. Introduces inert or nonreactive species into the system.
2. [reaction] ; [tag] = [power series]
 Tagged reaction is entered into system.
 $[reaction] = aA_{<s>} + bB_{<l>} + \dots = lL_{<l>} + mM_{<g>} + \dots$
 $[tag] = G, Ln(K), Lg(K), LN(K), LG(K).$
 $[power\ series] = \pm[constant], \pm[constant][label]$
 $[constant] = a, aE \pm b.$
 $[label] = *T*Lg(T), *T*Ln(T), *Lg(T), *Ln(T), /T, *T, *T**[integer], *T*LG(T), *T*LN(T), *LG(T), *LN(T).$
3. Carriage return. End of mnemonic reaction entry.

INPUT LIBRARY REACTION REFERENCE NUMBERS.

1. n. Include in the system the reaction whose library entry number is n.
2. $\pm n_1, \pm n_2 \dots$ Include in the system the reaction that is the algebraic sum of the reactions whose library entry numbers are $n_1, n_2 \dots$ (Total number of terms is limited to five.)
3. Carriage return. End individual library reference entry.

INPUT LIBRARY REACTION COLLECTION NUMBERS

1. n. Include in the group of reactions whose group library number is n. (n must be predefined in QPUT in a DATA declaration statement.)
2. Carriage return. End collective library reference entry.

INPUT STATE SPECIFICATIONS.

aA<s> + bB<l> + .. = lL<l> + mM<g> + ..

1. Carriage return. Leave state specification unaltered.
2. $s_2 + l + .. = l + g + ..$ Substitute state reaction for specification given, and tag the first solid as solid number 2.
3. $s_2 + l + .. \# l + g + ..$ Substitute state reaction for specification given, and tag the first solid as solid number 2. Because the # sign has replaced the = sign, tag the reaction as a reaction potential, i.e., do not include the reaction in the system.
4. #. Leave the state specification unaltered. Because the # sign has replaced the = sign, tag the reaction as a reaction potential, i.e., do not include the reaction in the system.
5. $s + L + .. = l + g + ..$ Substitute state reaction for specification given. Because the state designation for the first liquid is capitalized, remove first liquid from the system, i.e., null species B<L>.

ELEMENT LINKAGE INPUT?

1. Carriage return. No linkage present in problem.
2. YES(yes). Causes program jump to input linkage.
3. NO(no). No linkage present in problem.

(TYPICAL LINKAGE PROMPT)

1
He<g>

2
CO<g> = C<s> + 0.5O₂<g>

3
CO₂<g> = C<s> + O₂<g>

4
H₂O<g> = H₂<g> + 0.5O₂<g>

5
CH₄<g> = C<s> + 2H₂<g>

INPUT (E,I;n1,n2...).

1. C,1;2,3. Tag carbon with a 1 in reactions 2 and 3.
2. Carriage return. End linkage session.

Appendix A gives a typical session at the terminal in which a problem is run.

IX. SUBROUTINE QFIT

QFIT manages the system solver for QUIL. It also computes the errors and the error components in terms of the parameter set that is active. This expansion is only generated once for each iteration for use by the subroutine SYST. All entries into SYST during the current iteration will calculate using this expansion.

The code first updates the moles of elements, species, and phases that are present. It then proceeds to the calculation of the errors and their partials. For adsorbed species, control branches to a slightly more complex section of code.

After the function expansions are determined, QFIT branches to the subroutine SYST. An entry point into SYST that is used only by QFIT causes the function expansions to be stored in blank common by SYST. By appropriate methods that are described in Sec. XXIV, SYST computes a solution vector which is returned in SOLT.

Two preset limits are associated with the search strategy in QFIT. Both are used for comparison to the calculated jumps in the parameter set. The first is a coarse limit of the order of 0.2, and is associated with the branch to the direct search. The second is a fine limit of the order of parts per million, and is associated with the test for convergence. If the solution vector SOLT indicates a change in any of the parameters by as much as the coarse limit, control branches to DSER, which manages the direct search for QUIL. Else, if the solution vector SOLT requires a change in any of the parameters by as much as the fine limit, control branches to the direct jump with $t = 1.0$ and $r = 0.0$. Else, if the solution vector SOLT requires a change in no parameter by as much as the fine limit, the code compares the current value of the sum of the squares of the errors computed at the start of the current iteration to the value computed at the start of the last iteration. If the sum of the squares of the errors is monotone decreasing, control branches to the direct jump. Else, convergence is assumed.

The code has a running dump IDMP. This dump is a summary, or master, dump which is a single line of output containing numerical quantities and flags and which is printed once for each iteration. The number of the iteration is printed first, followed by a flag which is S or J, depending upon whether or not the parameters were altered as a result of a direct search or a direct jump. The determinant of the system is printed next, followed by a flag that is a blank, an asterisk, or an integer n, depending upon whether the system is normal, ill-conditioned, or of defect n. Next, the values of t , r , and the final sum of the squares are printed, each followed by a flag. The three flags indicate the progress of the system solver through the t search, the first semicircular t - r search, and the subsequent t - r searches at larger radii. The flag is a *, B, #, or \$, depending upon whether the search was aborted in the tripod search, bound in the tripod search, aborted in the parabolic search, or converged. A blank indicates that a search abort has occurred before the phase of the search in question. When the code is in the direct-search phase, the last quantity printed is the species requiring the largest relative change in value.

This dump is very informative, and should be active during execution.

X. SUBROUTINE SSQS(QERR)

The direct-search phase of QUIL requires the repetitive calculation of the sum of the squares of the errors. SSQS, the program name for QERRFIL, carries out these calculations as a function of the two direct-search parameters t and r .

The direct-search strategy is under program control. In the early stage of the direct search, the indefinite and the infinite operands occur frequently. If one attempts to reference such an

operand, an execution abort will result. Since the search strategy is designed to accept these occurrences as normal, each quantity must be tested after it is generated and before it is used to insure that it is a legitimate variable. If an infinite or indefinite operand is detected, the search algorithm branches and execution continues. This event is referred to here as a controlled abort.

Two logical variables are present upon return from SSQS to summarize the validity of the quantity returned as the sum of the squares of the errors. These two variables, EOK and POK, are returned as TRUE if the calculation is valid. The r parameter is applied in SOLV, which is a separate entry point in SYST. The t parameter is applied in SSQS. Consequently, abort situations in SOLV cause POK to be returned FALSE, and abort situations in SSQS cause EOK to be returned FALSE.

The routine first branches to SOLV. When control returns to SSQS, the solution vector is returned in the array SOLT. SSQS then computes the dummy array QQ from the current value of Q. (The dummy variable is introduced in order not to disturb the current position in parameter space, which is the array Q.) The code next computes the moles of elements, species, and phases present. Finally, the sum of the squares of the errors is computed with a section of code similar to that found in QFIT. For adsorbed species, control branches to a slightly more complex section of code.

The routine has a dump capability that is useful for debugging the search strategy in DSER. It should be activated with care, however, since it produces excessive output.

XI. SUBROUTINE APLT

The routine APLT plots the activity arrays that have been calculated in QUIL. The plot is made against whatever independent variable has been selected in the main program. One plot is generated for each phase present in the chemical environment.

The code begins by ordering the activity array against the independent variable. The code next converts the arrays to the negative logarithm of the activity for plotting. The lower-case characters are then converted to upper-case, since the plotter software makes no provision for the full ASCII convention. In this step, the state specification is removed in order to save space in the plot.

Next, the routine calculates the parameters necessary to label the plot. This labeling is necessary for readability, and entails tagging the arrays with the species Hollerith labels. Finally, the plotting is carried out, and the graph is tagged with whatever identifiers were input at the start of the first interactive session.

XII. SUBROUTINE RPLT

The routine RPLT plots any reaction potentials that may have been calculated by QUIL. The plot is made against whatever independent variable has been selected in the main program. One plot is generated for each reaction potential that has been calculated.

The routine begins by ordering the reaction potentials against the independent variable. The lower-case characters are then converted to upper-case, since the plotter software makes no provision for the full ASCII convention.

Finally, the plotting is carried out, and the graph is tagged with whatever identifiers were input at the start of the first interactive session.

XIII. PROGRAM FENG

The main program in FENG handles all of the library manipulation. It also executes all printed output. Input is from the keyboard, which is symbolic tape 1. Symbolic tape 2 contains the library. Printed output is to the terminal, which is symbolic tape 3. Plotter output is made to symbolic tape 5. Tape 6 is a scratch tape.

FENG is put into execution by a procedure file called FENG. This file gets symbolic tape 7 from a permfile called FENGDAT before FENG goes into execution. During execution, all numeric input for the problem is written to symbolic tape 8. After a normal exit from execution, the procedure file stores tape 8 as FENGDAT. The file FENGDAT may thus be regarded as the input data deck which is created during the interactive session with the user.

Numeric input may be corrected during execution in the event of a typographical error. However, typing mistakes in alphabetic input cannot be so corrected; the user must abort execution. Symbolic tape 8 contains all numeric input, so to rerun the problem, the operator simply replaces FENGDAT from tape 8 and executes the procedure file FENG again.

FENG can regenerate the documentation associated with any particular library entry. This capability is embedded in the main code FENG. If documentation for a particular fit is desired, the code reads the data from tape 2, and a duplicate of the original output is prepared.

When a new entry into the library is being made, FENG branches to the fitting subroutine FFIT. Upon return, the new entry into the library is made in accordance with the standard format. The output for the fit is printed, and execution is transferred to FPLT, where the fit is plotted.

FENG can output a library update before terminating execution. This printed output represents the current status of the library.

The prompts and responses for FENG are included below. The period (.) is not typed as part of the response.

INPUT REACTION IN FULL ASCII.

1. Carriage return. Causes program jump to output a reaction already in the library.
2. END. Terminates processing without library update.
3. end. Terminates processing with library update.
4. $aA<s> + bB<l> = cC<g>$. Initiates processing.

INPUT REACTION LIBRARY REFERENCE.

1. n. Outputs documentation for library entry n in standard format.

Appendix B gives a typical session at the terminal in which an entry is made into the library.

XIV. SUBROUTINE FPUT

FPUT is the subroutine in FENG that handles data input for the code, which must be in the full ASCII convention. If the numeric data is on symbolic tape 7, the program reads the input data from this tape.

The library is standardized on the SI system of units. To convert the input energies to joules, it is necessary to know the gaseous species coefficient for the overall reaction. The subroutine

CRAC, described in Sec. XXIII, is used to return the individual species coefficients, and the sum is determined in order to get the net coefficient.

The code next branches on the mode of input. If the input is in the form of the enthalpy and the entropy of reaction, control transfers to that section of the code where these data are input. In general, data are expected to be in the form of tables. FPUT can accommodate input as the equilibrium constant or as the free energy of reaction. Data may be in logarithmic or non-logarithmic form and the temperature may be in degrees Celsius or Kelvin. Output is standardized on the temperature in degrees Kelvin (K), the pressure in kilopascals (kPa), and the energy in kilojoules (kJ). As soon as the numeric data have been input, the data are written to tape 8.

The code then checks on library entry. ELIB has a library section and a bibliography section. A new library entry may cause a new entry in the library bibliography section or it may reference an old bibliography entry. In either case, the code obtains the information necessary to make or reference the entry.

If numeric data are from the keyboard, the code allows the operator to check the data. The routine prints all pertinent information, and then asks the user to check the data for accuracy. If errors exist in the data, the user can correct the numeric input during execution. If there is an error in the alphabetic input, the user must then abort. However, since the numeric input has been written to the backup tape 8, the user simply restores tape 8 to the permanent file FENGDAT. The procedure file, which executes FENG, copies tape 7 from FENGDAT during execution. The numeric data are thus available on tape 7 for tape input on subsequent runs.

The maximum number of parameters that may be used in the fit is reasonably large. Consequently, precision can be a real problem. Data are loaded into the library with 6-place accuracy. Parameters are loaded into the library with 12-place accuracy. To insure that reruns from the library are consistent with the fit as calculated on the first pass, all input is truncated to either 6- or 12-place accuracy before proceeding. Finally, the tables are ordered on the temperature.

The prompts and responses for FPUT are included below. The period (.) is not typed as part of the response.

DATA ON FILE?

1. YES(yes). Causes numerical input (only) from symbolic tape 7.
2. NO(no). Causes all input from keyboard.

TABULAR DATA?

1. YES(yes). Causes program jump to input tables.
2. NO(no). Causes program jump to input thermodynamic quantities.

INPUT NUMBER OF TABLES.

1. n. Causes program to set up to accept n tables of input. (n is limited to 4.)

INPUT TABULAR COEFFICIENTS.

1. $n_1, n_2, \dots$ Input one coefficient per table.

INPUT DATA AS (T(K),F(KCAL)).

1. T, F_1 , $F_2, \dots$ Input temperature and free energy.
2. Carriage return. Signals end of input list.

INPUT ENERGY AS (H,S).

1. H, S. Input enthalpy, entropy of reaction.

LIBRARY ENTRY?

1. YES(yes). Reaction is added to library.
2. NO(no). Reaction is not added to library.

INPUT A NEW BIBLIOGRAPHY REFERENCE IN FULL ASCII.

1. [reference]. Causes a new entry in the bibliography section. (May not exceed 10 lines.)
2. Carriage return. No new entry is made in the bibliography section.

INPUT AN OLD BIBLIOGRAPHY REFERENCE NUMBER.

1. n. References the fit to an old bibliography entry.

INPUT BIBLIOGRAPHY REPETITION FACTOR.

1. A(a). Tags library reference with a single alphabetic character to distinguish multiple library references to the same bibliography reference.

CHECK?

1. YES(yes). Causes execution to continue.
2. NO(no). Causes program jump to correct numerical data.

INPUT CORRECTIONS

1. n. T, F₁, F₂,... Retype lines that are in error.
2. Carriage return. Continue processing.

Appendix B gives a typical session at the terminal in which an entry is made into the library.

XV. SUBROUTINE FFIT

FFIT is the routine in FENG that manages the fitting procedure. The routine begins with 12 free parameters. If the number of data points is less than 13, the routine begins with one fewer parameters than there are data points. The fitting algorithm requires that all parameters as fitted be statistically significant. Here, the test of statistical significance is taken as the requirement that the parameter as fitted exceed by a factor of 2 the estimate of the standard deviation associated with the parameter in the fit. If any of the parameters in the fit are statistically insignificant, that parameter which is numerically the least significant is removed by nulling, and the fitting process is repeated.

The code begins by initializing all parameters. The maximum number of free parameters is fixed in accordance with the preceding paragraph. As the least-squares procedure is a linear fit to a power series, it is quite straightforward and will not be elaborated upon. The system of equations is solved in a linear system solver and the solution vector is truncated by an internal read/write with the same effective format as the read/write that takes place when the parameters are passed into the library. It is particularly important to truncate the parameters before calculating the variance of the fit, since the precision required in the 12-parameter fit may be quite high. After the variance is computed, the standard deviations for the various parameters may be computed.

Next, the code branches on the test for statistical significance. If any one of the parameters is insignificant, execution loops back through the subroutine. When convergence occurs, the maximum number of free parameters is set equal to the highest order parameter in the series that is nonzero. Because placement in the library is in the order of the series, the storage requirements for the library can thus be reduced.

XVI. SUBROUTINE FPLT

FPLT is the routine in FENG that plots the fitted parametric model. First the alphanumeric reaction must be tagged with the library entry number. This is done with encode/decode statements. In the process, the null bit 76B which flags the lower-case ASCII mode is removed, since the plotter does not have lower-case capabilities.

The free energy and the logarithm of the equilibrium constant are plotted over the temperature range of the data. Tabular data points are plotted for comparison.

XVII. PROGRAM SURF

The main program in SURF handles all of the library manipulation. It also executes all printed output. Input is from the keyboard, which is symbolic tape 1. Symbolic tape 2 contains the library. Printed output is to the terminal, which is symbolic tape 3. Plotter output is made to symbolic tape 5. Tape 6 is a scratch tape. The input to SURF is recorded on tapes 8 and 10 during execution. These tapes are stored by the procedure file that executes SURF. At the start of each run, the procedure file loads tapes 7 and 9 from the output of tapes 8 and 10. This looping facilitates the restart capabilities in SURF.

Because the parametric model embedded in SURF is a breakpoint model, many fitting attempts are required before a satisfactory fit results. It is assumed that tape 7 and tape 9 will be used extensively as input to restart each new fitting at the point where the last fitting ended. To facilitate this mode of operation, a NAMELIST input capability is provided to alter the parametric model associated with each run. In this mode of operation, tape 7 is used to input experimental data to the code, and tape 9 is used to input parametric data.

Experimental data input may be corrected during execution in the event of a typographical error. Typing mistakes in alphanumeric input require that the execution be aborted. However, the running updates to tapes 8 and 10 facilitate restart, since all input has been written to file (see Sec. XVIII).

SURF can regenerate the documentation associated with any particular library entry. This capability is embedded in the main code SURF. The original library entry is read from tape 2, and a duplicate of the original output is prepared.

When a new entry into the library is being made, SURF branches to the fitting subroutine SFIT. Upon return, the new entry into the library is made in accordance with the standard format. The output for the fit is printed, and execution is transferred to SPLT where the fit is plotted.

SURF also outputs a library update before terminating execution. This printed output represents the current status of the library. An update of the library can be obtained without executing a fit if the first two prompts from the code are answered with carriage returns.

The prompts and responses for SURF are included below. The period (.) is not typed as part of the response.

INPUT REACTION IN FULL ASCII.

1. Carriage return. Causes program jump to output documentation for a reaction already in the library.
2. END. Terminates processing without library update.
3. end. Terminates processing with library update.
4. A[s] = A<g>. Initiates processing. Reaction should be written so the species reaction coefficients are both 1.0.

INPUT REACTION LIBRARY REFERENCE.

1. Carriage return. Causes program jump to output a library update.
2. n. Outputs documentation for library entry n in standard format.

Appendix C gives a typical session at the terminal in which an entry is made into the library.

XVIII. SUBROUTINE SPUT

SPUT is the subroutine in SURF that handles data input for the code, which must be in the full ASCII convention. The code is put into execution with a procedure file which loads symbolic tape 7 from the permanent file SURFDAT and symbolic tape 9 from the permanent file SURFPAR. If no errors occur during execution, the procedure file will reload SURFDAT from tape 8 and SURFPAR from tape 10 before exiting execution. SURFDAT and SURFPAR therefore contain the current status of the breakpoint fit. After the code has been run once, the input for all subsequent runs with the same model can be input from tape. If the user specifies a different number of breakpoints, however, the new parametric model must be input from the keyboard.

The gaseous species coefficient for the overall reaction is not currently needed for the fitting in SURF. However, it is a part of the standard library format. Therefore the subroutine CRAC, described in Sec. XXIII, is used to return the individual species coefficients, and the sum is determined in order to get the net coefficient.

In general, experimental data input is expected to be in the form of tables. These data must be input as temperature, concentration or coverage, and pressure. Data may be in logarithmic or nonlogarithmic form, the temperature may be in degrees Celsius or Kelvin, and the pressure may be in atmospheres or kilopascals. Output is standardized on the SI system of units, where the temperature is in degrees Kelvin (K), the concentration in moles per square meter (mol/m^2), and the pressure in kilopascals (kPa). As soon as the experimental data have been input, they are written to tape 8.

Next, the code obtains from the user the details of the parametric model. Information associated with the breakpoint structure is input first. A logical array called MNULS controls which parameters in the model are free in the fit. These nulling parameters are input here. As the problem is nonlinear, some starting estimate of the parameters is required by the code. These estimates are also read at this time.

Conversion problems associated with units can be confusing and time-consuming. Consequently, SURF permits two modes of input. In the first, or graphical mode, the user answers questions that relate to a graph. In the second, or breakpoint mode, the user answers questions that relate to the breakpoint model. The graphical mode is usually more convenient on the first fitting attempt when the user has a plot or graph of the equilibrium data in hand. In this case, all questions must be answered in the system of units elected in the numeric phase of the interactive session. On subsequent fittings, the breakpoint input mode is undoubtedly to be preferred, since tape 9 will have the results of the last fitting in the standard units of the library.

If the parametric data have been input from tape 9, SPUT now gives the user a chance to reconfigure the parametric model for the current fit. All information is printed out relating to the model, and the code halts at separate NAMELIST inputs for the numerical coefficient arrays and for the logical nulling arrays. After the changes have been made, the code allows the user to check the arrays for accuracy. If an error exists, the code recycles through the NAMELIST check points until the data are accurate. As soon as the parametric data have been verified, the parametric data are written to tape 10.

The code then checks on library entry. ELIB has a library section and a bibliography section. A new library entry may cause a new entry in the library bibliography section or it may reference an old bibliography entry. In either case, the code obtains the necessary information to make or reference the entry.

If experimental data have been input from the keyboard, the code then allows the operator to check the data as they exist in storage. After the routine prints all of the pertinent information, the code asks the user to check the data for accuracy. If errors exist in the experimental data, the user can correct the data during execution. If there is an error in the alphabetic input, the user must then abort. During execution, the experimental data have been written to tape 8 and the parametric data have been written to tape 10. If no errors had occurred in execution, the procedure file would have replaced SURFDAT with tape 8 and SURFPAR with tape 10. To re-execute, the user should replace SURFDAT and SURFPAR with tape 8 and tape 10 and execute the procedure file again.

The maximum number of parameters that may be used in the fit is reasonably large; consequently, precision can be a real problem. Data are loaded into the library with 6-place accuracy. Parameters are loaded into the library with 12-place accuracy. To insure that reruns from the library are consistent with the fit as calculated on the first pass, all input is truncated to either 6- or 12-place accuracy before proceeding. Next, the tables are ordered on the temperature.

Several data arrays are unchanged throughout the fitting. These arrays are calculated before exiting SPUT. First, the number of distinct temperatures NTPS in the experimental data is determined and these temperatures are loaded into the array TVEC. Next, the limiting pressure PVEC associated with each data point is computed and loaded. Finally, the limiting pressure associated with each temperature is loaded into the array PZRO.

The prompts and responses for SPUT are included below. The period (.) is not typed as part of the response.

DATA ON FILE?

1. YES(yes). Causes experimental data input (only) from tape 7.
2. NO(no). Causes experimental input (only) from the keyboard.

INPUT DATA AS (T(K),C,P(kPa)).

1. Carriage return. Data format is not applicable.
2. T,C,P. Input tabular data as temperature, concentration or coverage, and pressure in units indicated.
3. Carriage return. Signals end of input.

INPUT DATA AS (T(K),log(C),log(P(atm))).

1. Carriage return. Data format is not applicable.
2. T,C,P. Input tabular data as temperature, concentration or coverage, and pressure in units indicated.
3. Carriage return. Signals end of input.

INPUT DATA AS (T(C),log(C),log(P(atm))).

1. Carriage return. Data format is not applicable.

2. T,C,P. Input tabular data as temperature, concentration or coverage, and pressure in units indicated.
3. Carriage return. Signals end of input.

INPUT DATA AS (T(C),C,P(atm)).

1. Carriage return. Data format is not applicable.
2. T,C,P. Input tabular data as temperature, concentration or coverage, and pressure in units indicated.
3. Carriage return. Signals end of input.

INPUT ADSORBATE PRODUCT SCALE FACTOR.

1. f. Input multiplicative scale factor to convert concentration or coverage to moles per square meter. If the concentration or coverage data were input as logarithmic quantities, input the logarithm of f.

INPUT SPECIFIC AREA FOR ADSORBENT.

1. A. Input specific area for adsorbent in square meters per mole. (Do not input logarithmic quantity.)

PARAMETERS ON FILE?

1. YES(yes). Causes program jump to read parametric model from symbolic tape 9.
2. NO(no). Causes program jump to input parametric model from keyboard.

INPUT NUMBER OF BREAKPOINTS.

1. M. Causes program to set up for a model with M breakpoints.

GRAPHICAL INPUT?

1. YES(yes). Causes program jump to input a graphical model.
2. NO(no). Causes program jump to input a breakpoint model.

INPUT BREAKPOINT TEMPERATURE.

1. T. Input an average temperature for the data in degrees Kelvin.

INPUT (C,P,F) FOR BREAKPOINT m.

1. C,P,F. Input a value of concentration or coverage, pressure, and the melding constant for breakpoint m. A typical first guess for a melding constant might be $F = 5$. Breakpoints are numbered from right to left.

INPUT (a,b,c,) FOR C1.

1. a,b,c. Input parameters a,b,c for the concentration at breakpoint 1. Breakpoints are numbered from right to left.

INPUT NULLS FOR C1.

1. T/F(t/f), T/F(t/f), T/F(t/f). Input nulling controls for (a,b,c) at concentration breakpoint 1. Breakpoints are numbered from right to left.

INPUT (a,b,c,d) FOR Rm.

1. a,b,c,d. Input parameters (a,b,c,d) for the ratio of breakpoint (m+1) to breakpoint m. Breakpoints are numbered from right to left.

INPUT NULLS FOR Rm.

1. T/F(t/f), T/F(t/f), T/F(t,f), T/F(t/f). Input nulling controls for (a,b,c,d) for the ratio of breakpoint (m+1) to breakpoint m. Breakpoints are numbered from right to left.

INPUT (a,b,c,d) FOR Sm.

1. a,b,c,d. Input parameters (a,b,c,d) for the slope between breakpoint m and breakpoint (m+1). Breakpoints are numbered from right to left.

INPUT NULLS FOR Sm.

1. T/F(t/f), T/F(t/f), T/F(t,f), T/F(t/f). Input nulling controls for (a,b,c,d) for the slope between breakpoint m and breakpoint (m+1). Breakpoints are numbered from right to left.

INPUT (a,b,c,d) FOR Fm

1. a,b,c,d. Input parameters (a,b,c,d) for the melding constant at breakpoint m. A typical first guess for a melding constant might be (a,b,c,d) = (0,3,0,1). Breakpoints are numbered from right to left.

INPUT NULLS FOR Fm.

1. T/F(t/f), T/F(t,f), T/F(t,f), T/F(t,f). Input nulling controls for (a,b,c,d) for the melding constant at breakpoint m. Breakpoints are numbered from right to left.

INPUT PRESSURE REFERENCE.

1. n. Input library reference number for the limiting pressure in the model, i.e., the library entry number of the solid or the liquid chosen as a reference state.

INPUT PARLIST(COEF,ROEF,SOEF,FOEF).

1. Carriage return. Leave PARLIST unchanged.
2. b\$PARLISTblistb\$. Input list of parameter changes for PARLIST. (See FORTRAN NAMELIST specifications.)
3. Carriage return. Signals the end of changes.

INPUT NULLIST(CNULS,RNULS,SNULS,FNULS).

1. Carriage return. Leave NULLIST unchanged.
2. `␣$NULLIST␣list␣`. Input list of control changes for NULLIST. (See FORTRAN NAMELIST specifications.)
3. Carriage return. Signals the end of changes.

LIBRARY ENTRY?

1. YES(yes). Reaction is added to library.
2. NO(no). Reaction is not added to library.

INPUT A NEW BIBLIOGRAPHY REFERENCE IN FULL ASCII.

1. [reference]. Causes a new entry in the bibliography section. (May not exceed 10 lines.)
2. Carriage return. No new entry is made in the bibliography section.

INPUT AN OLD BIBLIOGRAPHY REFERENCE NUMBER.

1. n. References the fit to an old bibliography entry.

INPUT BIBLIOGRAPHY REPETITION FACTOR.

1. A(a). Tags library reference with a single alphabetic character to distinguish multiple library references to the same bibliography reference.

CHECK?

1. YES(yes). Causes execution to continue.
2. NO(no). Causes program jump to correct numerical data.

INPUT CORRECTIONS.

1. n, T, C, P. Retype lines that are in error.
2. Carriage return. Continue processing.

Appendix C gives a typical session at the terminal in which an entry is made into the library.

XIX. SUBROUTINE SFIT

SFIT manages the fitting procedure for SURF. Because the model is a breakpoint model, the fit can become extremely nonlinear. Even when good data are available, the fitting procedure is very sensitive to the starting point in parameter space. To stabilize the fitting process, SFIT is so written that parameters are freed in the fit one at a time until all parameters that have not been nulled by the user are active. The first parameters to be freed are those associated with the slopes S_m . Then the breakpoint parameters C_1 and R_m are freed. The parameters F_m which control the asymptotic melding are the last to be added to the model. This stepwise approach increases the chances of locating a stable minimum for the model.

SFIT makes frequent use of equivalencing. All constants in the breakpoint model are computed from four parameters, except for C_1 which is computed from three parameters. For a model with M breakpoints, there is one value of C_1 , $(M-1)$ values of R , M values of S , and M values of F (see Sec. VI). This gives $3M$ constants in the breakpoint model. This structure is used to equivalence variables and arrays with the form $[(C,R),S,F]$. Such equivalencing results in compact coding structures and makes for a more readable code.

SFIT first initializes the control parameters. Then the routine frees one parameter at a time. At the reentry point, the code checks to see if all parameters freed by the user are active in the fit. If so, control exits the routine. If not, a new parameter is added to the group that is presently active, and the fitting procedure is repeated.

The fitting procedure is carried out in the log-log plane. Thus there are as many partial derivatives of error components with respect to the parameters at each data point as there are breakpoints in the model. For N data points and M breakpoints, we have

$$E_n = \sum_{m=1}^M e_m - \log(P_n); \quad n = 1, \dots, N. \quad (74)$$

where

$$e_m = S_m * \log \left(\frac{C_{m+1} \left(1 + u_{m+1}^{F_{m+1}} \right)^{G_{m+1}}}{C_m \left(1 + u_m^{F_m} \right)^{G_m}} \right), \quad (75)$$

and

$$e_M = S_M * \log \left(\frac{u_M}{\left(1 + u_M^{F_M} \right)^{G_M}} \right). \quad (76)$$

We adopt the convention that the partial of an error component with respect to a breakpoint constant, say S , is PERS. In particular, PERS(2,3) represents the partial of E_2 with respect to S_3 . The breakpoint constants are determined by the values of the parameters a , b , c , and d . We set $(a,b,c,d) = (1,2,3,4)$, and the partial of a breakpoint constant, say R_2 , with respect to the parameter d becomes PRRP(2,4). This convention is followed throughout.

SFIT uses a convergence test identical to that used by QFIT (see Sec. IX). When convergence occurs, SFIT solves for the standard deviation of the parameters in the fit. The code then truncates the parameters with the same precision that results when they are loaded into ELIB. Last, SFIT writes a copy of the fit to backup tape 10 to facilitate the rerun features of SURF.

SFIT has a running dump IDMP which is a summary, or master, dump. The format is identical to the running dump described in the subroutine QFIT (see Sec. IX). This dump is very informative, and should be active during execution.

XX. SUBROUTINE SSQS(SERR)

The direct-search phase of SURF requires the repetitive calculation of the sum of the squares of the errors. SSQS, which is the program name of SERRFIL, carries out these calculations as a function of the direct-search parameters t and r .

The direct-search strategy is under program control. In the early stage of the direct search, the indefinite and the infinite operands occur frequently. The search strategy is designed to accept these occurrences as normal. Since the direct-search package called by SURF is the same as that called by QUIL, the search philosophy in SSQS(SERR) is the same as that described in SSQS-(QERR) (see Sec. X).

The routine first branches to SOLV. When control returns to SSQS, the solution vector is returned in the array SOLT. SSQS then computes the dummy array POEG from the current value of POEF. (The dummy variable is introduced in order not to disturb the current position in parameter space, which is the array POEF.) The code then generates the breakpoint model from the parameter set that is currently active.

The routine has a dump capability that is useful for debugging the search strategy in DSER. It should be activated with care, however, because it produces excessive output.

XXI. SUBROUTINE SPLT

The routine SPLT plots the fit carried out in SURF. The plot is a log-log plot of pressure vs concentration.

The code begins by converting the label to upper-case and tagging the reaction with the library number assigned. The data points are then plotted, and the curves are drawn through the data. The curves are generated with the parametric model in two passes. The first pass is used to determine the plotting limits for the ordinate; the second pass generates the final data for the plot.

XXII. SUBROUTINE PAGE

PAGE controls paging for output from a terminal. The various options available are controlled by the calling argument CONT.

The routine requires a setup call that identifies the symbolic tape unit IPRT to which output will be directed. Also supplied is a four-character header HEADER, and two Hollerith variables containing the time TIME and date DATE.

A call to PAGE with control set to LASTER will cause a print of a 72-character string of asterisks.

A call to PAGE with CONT set to LSTART will start a new problem. The page number is set to 1, a header is printed, and two spaces are added after the header.

A call for a new page is carried out by setting CONT to LPAGE. The old page is finished out with blank lines, and a trailer is printed containing identification. The page number is augmented and a new header is printed.

In the normal mode of operation, a request to print n lines is made in the calling program by setting CONT = LPRINT and LINES = n. A call to PAGE causes the routine to check on the number of spaces available for printing on the current page. If space on the page is available, LINCNT is advanced n lines and control is returned to the calling program. If space is not available, the remainder of the page is filled with blank lines and a new page is restored. Clearly, the user may print from the calling program without a call to LPRINT if there is space available and the line counter LINCNT is advanced.

A call to finish a problem differs from a call for a new page only in that an additional asterisk row is printed. The call to PAGE with CONT = LABORT is also equivalent to a call to finish a problem.

The variable LINMAX controls the number of lines printed per page. To change the page length, therefore, change the value of LINMAX in the DATA declaration statement at the end of the code.

XXIII. SUBROUTINE CRAC

The subroutine CRAC is used to break down the chemical mnemonic reaction into the numerical quantities it represents. The routine is a multiple-pass processor, and relies entirely on encode/decode statements.

Input must be of the form

$$[\text{reaction}] ; [\text{tag}] = [\text{power series}] \quad (77)$$

A typical input might be

$$[\text{reaction}] = \text{H}_2\text{O}<g> = \text{H}_2<g> + 0.5\text{O}_2<g> , \quad (78)$$

where

$$[\text{tag}] = \text{Ln}(K) \quad (79)$$

and

$$[\text{power series}] = 1.0\text{E}-06 + 3.14*\text{Ln}(T) . \quad (80)$$

The routine first establishes a sentence array that delineates the various sections of the input sentence. The equal signs and the semicolon serve as delimiters for this step. The sentence is broken into four parts. The first part is the section of the chemical reaction to the left of the first equal sign. The second part is the section of the reaction from the right of the first equal sign to a blank or a semicolon, whichever comes first. Species located in the first and second parts of the sentence will have species coefficients that differ in sign. The third part of the input sentence is the identifier for tagged input, and is located between the semicolon and the next equal sign. The fourth and last part is the section that follows the second equal sign. This is the section containing the energy or equilibrium constant data as a power series in the temperature T to which may have been added terms in Log(T) and/or Ln(T).

The routine next establishes a word array from the input sentence. To do this, it parses the sentence by branching on the plus, minus, and equal signs in the reaction. A leading plus sign is assumed at the start of sections 1, 2, and 4.

The code next determines the parametric form of the tagged input, if any such input has been included. Input must be labeled G, Ln(K), Lg(K), LN(K), or LG(K). The numeric coefficients may be in the E or the F format. Each coefficient is decoded in order, and the parameter label is extracted. This label may be missing (a constant), or else it must be *T*Lg(T), *T*Ln(T), *Lg(T), *Ln(T), /T, *T, *T**, *T*LG(T), *T*LN(T), *LG(T), or *LN(T). This label determines where the decoded coefficient is loaded into the parameter array.

Next, the routine determines those species that are present. To do this, it branches on the word array and discards leading coefficients. In the process, the species collected are checked for uniqueness. As a part of this step, the states of the species that have been found are determined. In this determination, the convention is adopted that triangular brackets <> denote macroscopic

phases and that square brackets [] denote surface phases. A state number s ranging from 1 to 6 is assigned each species depending upon whether it is solid, liquid, or gas, or whether it is adsorbed on a solid, liquid, (or gas). These six possibilities are coded into a state identifier S treated modulo 6. Thus, any species has a unique state identifier

$$S = s + 6*(i-1) , \quad (81)$$

where i is the phase number. In addition, if a state specification is capitalized, it is removed from the computational system by nulling.

Next, the elements that are present are determined by scanning the species present. Here, the element tags resulting from linkage are retained. To determine the elements present, two arrays are established—a character array and a flag array. The flag array is generated from the character array and is used as a parsing guide to the sentence. An equivalent array is used in the determination of the element coefficients. The use of the flag array in determining the element coefficients is more complicated; therefore, in the interest of clarity, the description of the flag array is postponed.

Next, the species coefficients are determined. These coefficients are the factors preceding the species mnemonics. They are picked off by scanning the word matrix that was computed earlier, and decoding the character string containing the coefficient.

To determine the elemental coefficients, three arrays are established—a character array, a flag array, and a coefficient array. These three arrays are generated in the order given. Furthermore, each location in each array is related to and derived from the preceding array.

The elements themselves are composed of upper- and lower-case alphabetic characters. The lower-case character differs from the upper-case character in that it is preceded by a null character 76B. The elements may also be tagged with / n /, where n is a single integer number.

The flag string is loaded with a running index that is generated from the character string. The running index is initialized to zero, and a scan begins of the character array. The characters are processed one at a time. If a new character is alphabetic, the running index is augmented by one. If the new character is the null bit 76B, the index is diminished by one. For each successive location that is processed in the character string, an entry is made into the flag string. If the character in the character string is part of an element designation, the entry into the flag string is the running index. If the character in the character string is not part of an element designation, a -1 is loaded into the flag string.

The flag string is then reset in the following way: the flag string is scanned from left to right, and each positive entry in the string is replaced by the largest number thus far encountered in the scan. This results in a monotone increasing string for positive numbers. It should be noted that positive plateaus in the flag string now correspond to single element regions in the character string. Moreover, the number in the flag string is the ordinal of the element in the species, and the largest number in the string is the cardinal of elements in the species. A final pass through the flag string sets all negative numbers to zero except where corresponding locations in the character string contain a left or right parenthesis.

A coefficient string is then established. The element constant is the number—real or implied—following the element designation. A parenthetic constant is the number—real or implied—that follows any right parenthesis. The element coefficient is the product of the element constant with any parenthetic constant(s). To load the coefficient string, several passes are made through the character string. On the first pass, element constants are decoded and loaded into the coefficient string in locations corresponding to the element locations in the character string. On the second pass, parenthetic constants are decoded and loaded into the coefficient string in locations corresponding to the parenthetic locations in the character string, except that they are distinguished from element constants by being loaded as negative numbers. A third pass is made in

which the constants are cascaded by multiplying element constants with parenthetic constants, where parentheses are present. The coefficient string now contains element coefficients in locations analogous to the locations in the character string that contain the elements. The arrays are packed by removing extraneous locations not containing element designations. By a scan of the flag array, the character array can be decomposed into elements whose coefficients are located in analogous positions in the coefficient array.

Before exiting CRAC, a check is made to determine that the reaction being processed is stoichiometrically self-consistent, i.e., to insure that matter is conserved in the overall reaction. If species coefficients are present that do not terminate in their binary representation, the check will fail due to roundoff error.

A dump capability exists in CRAC. By setting the logical variable DMP in the DATA declaration to TRUE, the user can get an output listing at the end of each subroutine entry.

XXIV. SUBROUTINE SYST

The routine SYST manages the system solver for QUIL. SYST has three entry points—SYST, SOLV, and SIGM—and requires six major arrays. There are two working matrices, AM and BM, two scratch matrices, AS and BS, and two storage matrices, SA and SB. The solution vector is returned in the array SOLT.

SYST requires a setup call under the entry point SYST to initialize the routine for subsequent working calls. The Taylor's series expansions have been computed in QUIL and loaded into the working arrays, AM and BM. Upon entry into SYST, these arrays are stored in SA and SB. SYST first attempts a solution by the Gauss-Newton method. If precision is lost in this method beyond an acceptable limit, the code reverts to the Newton-Raphson method. In either case, SOLT(n,1) is loaded with the calculated solution for the unperturbed system, i.e., the solution for $r = 0.0$.

When a solution vector is required by QUIL, a call is made to the second entry point SOLV with r as a calling argument. SOLV calculates a new solution by whatever method has been selected under the SYST call and returns the solution in SOLT(n,2). If the Newton-Raphson method is in use and the system is singular, the solution is minimized in a least-squares sense against the homogeneous solution, and the homogeneous solution is returned in the scratch matrix AS.

Least-squares problems frequently require calculation of the cross- and covariances. To facilitate such calculations, an entry point SIGM has been provided to return the inverse of the system matrix in the working array AM.

The routine has various entry/exit dump capabilities which can be activated with the logical variables MDMP, YDMP, ODMP, and IDMP. These variables are found in the SYST DATA declaration statements.

XXV. SUBROUTINE DSER

DSER manages the direct search for QUIL (see Sec. III). It relies on two subroutines, TRIP and PARA, that execute the tripod and parabolic searches. The routine also calls a subroutine SSQS that returns the sum of the squares of the errors ϕ for a given point in the t - r plane. The code is independent of the least-squares model in the calling program, and operates in a setup phase, a t search phase, and a t - r search phase.

The routine begins by getting a bound on t . The value of t is increased until a program abort is detected in SSQS or until a value of ϕ is returned that is larger than the initial value at the origin QINT. DSER then checks for a negative slope at the origin, because a positive slope at the

origin will defeat the search strategy. If a positive slope is found, DSER returns control to the calling program with the logical variable AOK set to FALSE. If a negative slope is found, DSER exits the setup phase.

The code then proceeds to locate the value of t corresponding to the minimum in the sum of the squares of the errors ϕ on the t axis. It does so in two phases: a tripod search and a parabolic search. The tripod search is conducted in TRIP and the parabolic search is conducted in PARA. These two routines search for extrema. The control parameter, IXTRM, is set to a -1 to cause a search for a minimum and to a $+1$ to cause a search for a maximum. Both routines return a logical variable NORM; if the exit was normal, NORM is set to TRUE. In addition, the calling arguments to TRIP include the permissible search range, and a logical variable LBND which is returned as TRUE by TRIP if the search ended with a monotone decreasing ϕ surface located against a bound, i.e., a controlled abort situation.

The search algorithm next requires that the search proceed into the t - r plane. The code has now determined the minimum on the t axis (t_1 , Q_1) and a radius r_1 . The slope of the ϕ surface into the t - r plane is highly variable from iteration to iteration, varying by many orders of magnitude. The search strategy thus normalizes the p coordinate so that the variation around the semicircle is of the same order as the variation across the semicircle in the t direction. To do this, we replace r by fr , where the factor f is so determined that

$$\frac{\partial \phi}{\partial t} \sim \frac{\partial \phi}{\partial (fr)} \quad (82)$$

The search next proceeds into the t - r plane. Pathological problems frequently have multiple troughs emanating from the minimum at t_1 . Typically, two troughs are found at ± 85 - 88 degrees from the r axis. The first search proceeds over the entire semicircle and seeks out all maxima and minima. The location of the deepest trough so located is saved.

DSER then proceeds to search larger semicircles along this trough. When systems are particularly nonlinear, deep troughs will occur at angles as large as 89.9 degrees. After the search at each new radius—if the angle of the minimum exceeds 45 degrees—the f factor is now adjusted so that the trough will occur near 45 degrees on the next semicircular search. This also tends to flatten the trough and simplify the search. When the new minimum found at the larger radius is larger than the old minimum found at the smaller radius, the semicircular search is abandoned.

At all times during an iteration, the variables QSOL, ESOL, and PSOL contain the values of the lowest minimum thus far located. After each calculation of ϕ , the new value is compared to the stored value and, if lower, replaces the lowest minimum as the best solution thus far located. Therefore, the normal exit from the subroutine is also the abort exit; the final value of ϕ QFNL is loaded from QSOL.

XXVI. SUBROUTINE TRIP

Subroutine TRIP conducts the tripod search for DSER (see Sec. III). The calling arguments to TRIP specify an interval, XLOW to XHGH, and return two logical variables, LBND and NORM. LBND is returned TRUE if the search has ended with a monotone decreasing ϕ surface blocked by a bound, i.e., a controlled abort situation. NORM is returned TRUE if the exit from TRIP was normal. TRIP will search for a maximum or a minimum, depending upon whether the control parameter IXTRM is $+1$ or -1 .

After initializing the routine, TRIP computes parameters that control the search algorithm. The code searches for extrema by manipulating three equidistant points along the search trajectory. The location and spacing of the points constitute the search variables. The search algorithm

provides that the three points may be moved in three ways: they may be jumped, stepped, or merged. In the jump move, the three points are moved left or right in such a way that the outer two locations are retained in the triad and the spacing doubles. In the step move, the two leftmost or rightmost locations are retained in the triad and the spacing remains fixed. In the merge move, the leftmost or rightmost locations are retained in the triad but the spacing is halved.

As a result of each iteration, one calculated value of the sum of the squares of the errors is discarded. The value for the new location needs to be recalculated on each pass, and the logical array `VALID` contains the information as to which location needs to be recalculated.

The code then branches on the calculation of `r` and `t`, depending upon whether `DSER` is in the `t` search or the `t-r` search. The values computed here are checked for an extremum (see Sec. XXV). If a controlled abort situation has been detected in `SSQS`, the upper bound `XHGH` is reset, the control arrays are back-loaded from buffer storage, and control branches to the test section. Else, the values for the current triad are loaded into buffer storage.

The test for convergence follows. The standard deviation of `phi` is calculated for the three locations, and convergence is taken when the normalized standard deviation is less than 0.01.

The control parameters are then computed. A test is made for local and/or global maxima and/or minima. If gains have been made in `TRIP` and bounding has not occurred, `XMIN` is set `TRUE` to flag the fact that execution is normal. A test then follows to check for oscillation between regions of normal and inverted curvature. The integer `ISSET` counts the number of times the search enters a region with curvature of the opposite sense from that required by the extremum search; if 10 such reversals take place, a controlled abort results.

The jump parameter `IX` is then computed from the control variables. `IX` = 1, 2, or 4 as the move is a merge, step, or jump; the sign of `IX` determines the direction of the move.

`TRIP` then tests for exit conditions. If convergence has not occurred, a move is taken. Else, if the triad is centered on an extremum, a normal exit occurs. Else, if a global extremum has been found, control branches to a move. Else, if `XMIN` is `TRUE`, control branches to a move. Else, if the search is bounded, a normal exit is taken. Else, a move is taken.

XXVII. SUBROUTINE PARA

Subroutine `PARA` conducts the parabolic search for `DSER` (see Sec. III). The calling arguments to `PARA` specify the arrays for the direct search and provide for a logical variable `NORM`, which is returned `TRUE` if the exit from `PARA` is normal. `PARA` will search for a maximum or a minimum, depending upon whether the triad of points delivered to `PARA` by the calling arguments spans a maximum or a minimum.

After initializing the routine, `PARA` proceeds to calculate the new jump. First, `PARA` fits a parabola to the three points used by the search. In this fit, various tests are made on the validity of the calculation, and abort situations are anticipated.

The code branches on the calculation of `r` and `t`, depending upon whether `DSER` is in the `t` search or the `t-r` search. The values computed here are checked for an extremum (see Sec. XXV). If a controlled abort is detected in `SSQS`, control is returned to the calling program after a test for a normal exit.

`PARA` now has values for the sum of the squares of the errors computed at four locations along the search trajectory. To advance the search algorithm, three points are selected from the set of four. The locations selected are the extremum of the four and the nearest neighbor on either side. These three points always span the extremum, and the algorithm is stable.

Finally, `PARA` tests for convergence. If fewer than four passes have been made in `PARA`, control cycles back for another pass. Else, if the new value of the sum of the squares of the errors

represents an improvement on the value stored as the present extremum, control cycles back for another pass. Else, control returns to the calling program with a normal exit.

XXVIII. PROGRAM LREF

Reactions can be entered into QUIL individually or in collections. To enter collections of reactions, the collections must be predefined in a QPUT DATA declaration statement. As a convenience feature for the user, a code LREF will list all collections that have been predefined. The program LREF anticipates that this declaration will follow a particular format. If one wishes to predefine the three library entries 2, 7, and 45 as collection number 4, a DATA declaration statement

```
DATA LIBL(4), [LIBDUM(I), I=31,33]/3,2,7,45/ (83)
```

would be required. This format must be followed exactly if LREF is to be employed.

The code is put into execution with a procedure file LREF which gets the library ELIB and the subroutine QPUT. The code first reads the DATA statements in QPUT to get the appropriate collection lists. Using these lists, the code then proceeds to tabulate and cross-reference the various collections by editing the entries in ELIB. LREF uses PAGE to write the output to a terminal.

APPENDIX A

SAMPLE SESSION WITH QUIL

A typical session is outlined wherein a calculation is made on the chemical environment in the coolant of the gas-cooled reactor. The stoichiometry is typical of steady-state operation and includes a cesium sorption isotherm in the calculation.

In this particular example, gas-solid equilibrium has been inhibited. Solid carbon is introduced as a separate reaction, and is tagged to prevent gas-solid interaction. This permits the cesium present to react with the solid to form surface phases (see Sec. VIII). In addition, the carbon reaction potential is carried along in the calculation.

The actual prompts and responses are indicated below. The procedure file QUIL puts the code QUILCOD into execution.

```
old, quil  
READY.  
ascii  
READY.  
-quil
```

```
INPUT RUN COMMENTS.  
? PEACH BOTTOM STOICHIOMETRY.  
? TEST EXAMPLE.
```

PEACH BOTTOM STOICHIOMETRY.
TEST EXAMPLE.

CHECK?

? yes

INPUT REACTIONS IN FULL ASCII.

? He<g>

? C<s>

?

INPUT LIBRARY REACTION REFERENCE NUMBERS.

? 2

? 17

?

INPUT LIBRARY REACTION COLLECTION NUMBERS.

? 2

? 7

?

INPUT STATE SPECIFICATIONS.

$\text{CO}_2\langle g \rangle = \text{C}\langle s \rangle + \text{O}_2\langle g \rangle$

? g#S+g

$\text{Cs}\langle s \rangle = \text{Cs}\langle g \rangle$

?

$\text{H}_2\text{O}\langle g \rangle = \text{H}_2\langle g \rangle + 0.5\text{O}_2\langle g \rangle$

?

$\text{CO}_2\langle g \rangle = \text{CO}\langle g \rangle + 0.5\text{O}_2\langle g \rangle$

?

$\text{CO}\langle g \rangle + 2\text{H}_2\langle g \rangle = \text{CH}_4\langle g \rangle + 0.5\text{O}_2\langle g \rangle$

?

$\text{Cs}(\text{OH})\langle g \rangle = \text{Cs}\langle g \rangle + 0.5\text{O}_2\langle g \rangle + 0.5\text{H}_2\langle g \rangle$

?

$\text{CsO}\langle g \rangle = \text{Cs}\langle g \rangle + 0.5\text{O}_2\langle g \rangle$

?

$\text{Cs}_2\langle g \rangle = 2\text{Cs}\langle g \rangle$

?

$\text{Cs}_2(\text{OH})_2\langle g \rangle = 2\text{Cs}\langle g \rangle + \text{O}_2\langle g \rangle + \text{H}_2\langle g \rangle$

?

$\text{Cs}_2\text{O}\langle g \rangle = 2\text{Cs}\langle g \rangle + 0.5\text{O}_2\langle g \rangle$

?

ELEMENT LINKAGE INPUT?

? yes

1

He<g>

²
C_s>

³
CO₂_g#C_S>+O₂_g;[2]

⁴
Cs[_s]=Cs_g;[17]

⁵
H₂O_g=H₂_g+0.5O₂_g;[1]

⁶
CO₂_g=CO_g+0.5O₂_g;[46]

⁷
CO_g+2H₂_g=CH₄_g+0.5O₂_g;[48]

⁸
Cs(OH)_g=Cs_g+0.5O₂_g+ 0.5H₂_g;[12]

⁹
CsO_g=Cs_g+0.5O₂_g;[13]

¹⁰
Cs₂_g=2Cs_g;[14]

¹¹
Cs₂(OH)₂_g=2Cs_g+O₂_g+ H₂_g;[15]

¹²
Cs₂O_g=2Cs_g+0.5O₂_g;[16]

INPUT (E,I;N1,N2...)
? C,1;2
?

TEMPERATURE(K)?
? 1400, 1000, -50

PRESSURE(kPa)?
? 2296

MOLES He_g
? 1000

MOLES C/1/_s>
? 6000

MOLES CO₂<g>
?

MOLES C<S>
?

MOLES O₂<g>
?

MOLES Cs[s]
? 7.2E-04

INPUT SPECIFIC AREA FOR Cs[s]
?

MOLES Cs<g>
?

MOLES H₂O<g>
? 0.0005

MOLES H₂<g>
? 0.009

MOLES CO<g>
? 0.0005

MOLES CH₄<g>
? 0.0006

MOLES Cs(OH)<g>
?

MOLES CsO<g>
?

MOLES Cs₂<g>
?

MOLES Cs₂(OH)₂<g>
?

MOLES Cs₂O<g>
?

ELEMENTS OF INTEREST?
? yes

M ELEMENTS

1	He
2	C/1/
3	C
4	O
5	Cs
6	H

INPUT M.

? 5

?

NUMBER OF OUTPUT DATA SETS?

?

READY

Both the plotter and the printer outputs are written to file. For the initial calculation, QUIL outputs the stoichiometry conditions for which the computation was initiated, and then prints the equilibrium conditions for the environment. Output for all subsequent calculations resulting from "do loops" will contain only the resultant chemical equilibria. However, if the stoichiometry is under looping control, the stoichiometry conditions are output for each new calculation.

Output for this run is included in this section. The keyboard responses indicated above caused output to be written for each temperature calculated. Only the first output temperature is included, along with the plotted equilibrium distributions.

QUIL
15.10.05

QUIL
76/09/07

PAGE 1.

PEACH BOTTOM STOICHIOMETRY.
TEST EXAMPLE.
PRESSURE AT 2.296E+03 kPa.

SYSTEM

N	LN#	REACTION
1	F	He<g
2	T	C<1/<g
3	F	CO2<g =C<1 +O2<g ;[2]
4	F	C<[s]=C<g ;[17]
5	F	H2O<g =H2<g +0.5O2<g ;[1]
6	F	CO2<g =CO<g +0.5O2<g ;[46]
7	F	CO<g +2H2<g =CH4<g +0.5O2<g ;[48]
8	F	C<[OH]<g =C<g +0.5O2<g +0.5H2<g ;[12]
9	F	C<[O]<g =C<g +0.5O2<g ;[13]
10	F	C<[s]<g =2C<g ;[14]
11	F	C<[s]<[OH]<2<g =2C<g +O2<g +H2<g ;[15]
12	F	C<[s]<[O]<g =2C<g +0.5O2<g ;[16]

LINKS

L	E	TAG	REACTIONS
1	C	1	2

SURFACES

N	A0	REACTION
4	6.00600E+00	C<[s]=C<g ;[17]

SPECIES

K	MOLES	SPECIES
1	1.00000E+03	He<g
2	6.00000E+03	C<1/<g
3	0.	CO2<g
4	0.	C<[s]
5	0.	O2<g
6	7.20000E-04	C<[s]
7	0.	C<[g]
8	5.00000E-04	H2O<g
9	9.00000E-03	H2<g
10	5.00000E-04	CO<g
11	6.00000E-04	CH4<g
12	0.	C<[OH]<g

-IDENTIFICATION-

OUIL
15.10.05

PAGE 2.

OUIL
76/09 07

SPECIES

K	MOL	SPECIES
13	0.	C≡O'g'
14	0.	C≡2'g
15	0.	C≡2'OH'2'g
16	0.	C≡2O'g'

ELEMENTS

M	NUL	MOL	ELEMENT
1	F	1.00000E+03	He
2	F	6.00000E+03	C 1
3	F	1.10000E-03	C
4	F	1.00000E-03	O
5	F	7.20000E-04	C≡
6	F	2.14000E-02	H

-IDENTIFICATION-

QUIL
15.10.05

QUIL
76/09 07

PAGE 3.

CONDITIONS

TEMP (K)	PRES (kPa)
1.40000E+03	2.29600E+03

SYSTEM

N	NUL	K	REACTION
1	T		He(g)
2	T		C(1)/s
3	T	1.63745E-15	CO2(g) = C(1) + O2(g) ; [2]
4	F	1.45164E+03	Cs(s) = Cs(g) ; [17]
5	F	4.51954E-06	H2O(g) = H2(g) + 0.5O2(g) ; [1]
6	F	9.73910E-06	CO2(g) = CO(g) + 0.5O2(g) ; [4b]
7	F	7.04641E-15	CO(g) + 2H2(g) = CH4(g) + 0.5O2(g) ; [48]
8	F	1.12167E-06	Cs(OH)(g) = Cs(g) + 0.5O2(g) + 0.5H2(g) ; [12]
9	F	4.30484E+01	CsO(g) = Cs(g) + 0.5O2(g) ; [13]
10	F	7.25978E+03	Cs2(g) = 2Cs(g) ; [14]
11	F	8.42182E-09	Cs2(OH)2(g) = 2Cs(g) + O2(g) + H2(g) ; [15]
12	F	8.87149E+00	Cs2O(g) = 2Cs(g) + 0.5O2(g) ; [16]

PHASES

J	NUL	STATE	NUM	MOLS
1	F	GAS	1	1.00000E+03
2	F	SOL	1	6.00000E+03
3	T	SOL	0	
4	F	ADS	1	7.19880E-04

SPECIES

K	NUL	STATE	MOLS	ACT	J	SPECIES
1	F	GAS	1.00000E+03	2.29597E+03	1	He(g)
2	F	SOL	6.00000E+03	1.00000E+00	2	C(1)(s)
3	F	GAS	4.20496E-15	9.65447E-15	1	CO2(g)
4	T	SOL			3	C(s)
5	F	GAS	7.30456E-34	1.67711E-33	1	O2(g)
6	F	ADS	7.19880E-04	1.99767E-08	4	Cs(s)
7	F	GAS	1.20381E-07	2.76391E-07	1	Cs(g)
8	F	GAS	9.51426E-14	2.18445E-13	1	H2O(g)
9	F	GAS	1.05000E-02	2.41077E-02	1	H2(g)
10	F	GAS	1.00000E-03	2.29597E-03	1	CO(g)
11	F	GAS	1.00000E-04	2.29597E-04	1	CH4(g)
12	F	GAS	6.82416E-19	1.56681E-18	1	Cs(OH)(g)
13	F	GAS	1.14520E-25	2.62934E-25	1	CsO(g)
14	F	GAS	4.58309E-18	1.05226E-17	1	Cs2(g)
15	F	GAS	1.59732E-40	3.66741E-40	1	Cs2(OH)2(g)

-IDENTIFICATION-

 QUIL
 15.10.05

 QUIL
 76/09 07

PAGE 4.

SPECIES

K	NUL	STATE	MOLES	ACT	J	SPECIES
16	F	GAC	1.53591E-31	3.52640E-31	1	Cs2O<g>

REACTION POTENTIAL(S)

N	P	REACTION
1	3.97433E+00	CO2<g>+C S'+O2<g>+ [2]

ELEMENT(S) OF INTEREST

Cs

K	SPECIES MOLES	ELEMENTAL MOLES	PEP CENT	SPECIES
6	7.19880E-04	7.19880E-04	99.98	Cs[cs]
7	1.20381E-07	1.20381E-07	.02	Cs'<g>
12	6.82416E-19	6.82416E-19	0.00	Cs'OH'+<g>
13	1.14520E-25	1.14520E-25	0.00	CsO <g>
14	4.58309E-18	9.16618E-18	0.00	Cs2'<g>
15	1.59732E-40	3.19464E-40	0.00	Cs2'OH'+2'<g>
16	1.53591E-31	3.07182E-31	0.00	Cs2O <g>
		7.20000E-04	100.00	

-IDENTIFICATION-

Fig. A-1.
Phase 1<gas>. Pressure at $2.296E+03$ kPa. Peach Bottom stoichiometry. Test example.

Fig. A-2.
Phase 2<sol>. Pressure at $2.296\text{E}+03$ kPa. Peach Bottom stoichiometry. Test example.

Fig. A-3.
 Phase 3<ads>. Pressure at $2.296E+03$ kPa. Peach Bottom stoichiometry. Test example.

Fig. A-4.
Phase 1<gas>. Pressure at 2.296E+03 kPa. Peach Bottom stoichiometry. Test example.

Fig. A-5.
 Phase 4<ads>. Pressure at 2.296E+03 kPa. Peach Bottom stoichiometry. Test example.

Fig. A-6.

Phase 1<gas>. Pressure at $2.296\text{E}+03$ kPa. Peach Bottom stoichiometry. Test example.

APPENDIX B

SAMPLE SESSION WITH FENG

A typical session is outlined wherein a reaction is loaded into the library using FENG. The code is put into execution with a procedure file. Once in execution, the code will ask the user for the necessary information to carry out the fit.

Data in the library represent the standard free energy of reaction. The convention is adopted that, where possible, a reaction be written in terms of standard states. Consider the reaction involving $\text{H}_2\text{O} <g>$. To enter this species into the library with the associated thermodynamics, we first write the reaction desired.

Here ΔG_r is the free energy of reaction. For the reactions of formation, we may write

and

where $<r>$ indicates the reference state, and ΔG_r is the free energy of formation for the species in question. Note that we can write

$$-1.0*(\text{Eq. I}) + 1.0*(\text{Eq. II}) + 0.5*(\text{Eq. III}), \quad (\text{B-5})$$

and thereby derive Eq. (B-1). Similarly, we can write

$$\Delta G_r = -1.0*\Delta G_f^{\text{I}} + 1.0*\Delta G_f^{\text{II}} + 0.5*\Delta G_f^{\text{III}}, \quad (\text{B-6})$$

and thereby derive the free energy of reaction for Eq. (B-1). As input to FENG, we will therefore need the three tables of free energies of formation and the three coefficients

$$c_i = -1.0, +1.0, +0.5; i = 1, 2, 3. \quad (\text{B-7})$$

The actual prompts and responses are indicated below. Data are taken from JANAF.⁶ The procedure file FENG puts the code FENGCD into execution.

```
old,feng
READY.
ascii
READY.
-feng
```

INPUT REACTION IN FULL ASCII.

? $\text{H}_2\text{O}_{<\text{g}>} = \text{H}_{2<\text{g}>} + 0.5\text{O}_{2<\text{g}>}$

DATA ON FILE ?

? no

TABULAR DATA?

? yes

INPUT NUMBER OF TABLES.

? 3

INPUT TABULAR COEFFICIENTS.

? -1.0,+1.0,0.5

INPUT ENERGY DATA AS (T(K),F(KCAL)).

? 300,-54.617,0,0

? 400,-53.519,0,0

? 500,-52.361,0,0

? 600,-51.156,0,0

? 700,-49.915,0,0

? 800,-48.646,0,0

? 900,-47.352,0,0

? 1000,-46.040,0,0

? 1100,-44.712,0,0

? 1200,-43.371,0,0

? 1300,-42.022,0,0

? 1400,-40.663,0,0

? 1500,-39.297,0,0

?

LIBRARY ENTRY?

? yes

INPUT A NEW BIBLIOGRAPHY REFERENCE IN FULL ASCII.

? JANAF Thermochemical Tables

? The Dow Chemical Co.

? Midland, Michigan

?

INPUT BIBLIOGRAPHY REPETITION FACTOR.

? a

$\text{H}_2\text{O}_{<\text{g}>} = \text{H}_{2<\text{g}>} + 0.5\text{O}_{2<\text{g}>}$

TABLE(S)=3

	COEF(S)=	-1.00000E+00	1.00000E+00	5.00000E-01
1	3.00000E+02	-5.46170E+01	0.	0.
2	4.00000E+02	-5.35190E+01	0.	0.
3	5.00000E+02	-5.23610E+01	0.	0.
4	6.00000E+02	-5.11560E+01	0.	0.
5	7.00000E+02	-4.99150E+02	0.	0.
6	8.00000E+02	-4.86460E+01	0.	0.
7	9.00000E+02	-4.73520E+01	0.	0.
8	1.00000E+03	-4.60400E+01	0.	0.
9	1.10000E+03	-4.47120E+01	0.	0.
10	1.20000E+03	-4.33710E+01	0.	0.
11	1.30000E+03	-4.20220E+01	0.	0.
12	1.40000E+03	-4.16630E+01	0.	0.
13	1.50000E+03	-3.92970E+01	0.	0.

JANAF Thermochemical Tables
The Dow Chemical Co.
Midland, Michigan

REPETITION FACTOR A.

CHECK?

? no

INPUT CORRECTIONS.

? 12,1400,-4.06630E+01,0,0

?

$H_2O_{<g>} = H_2_{<g>} + 0.5O_{2<g>}$

TABLE(S)=3

	COEF(S)=	-1.00000E+00	1.00000E+00	5.00000E-01
1	3.00000E+02	-5.46170E+01	0.	0.
2	4.00000E+02	-5.35190E+01	0.	0.
3	5.00000E+02	-5.23610E+01	0.	0.
4	6.00000E+02	-5.11560E+01	0.	0.
5	7.00000E+02	-4.99150E+01	0.	0.
6	8.00000E+02	-4.86460E+01	0.	0.
7	9.00000E+02	-4.73520E+01	0.	0.
8	1.00000E+03	-4.60400E+01	0.	0.
9	1.10000E+03	-4.47120E+01	0.	0.
10	1.20000E+03	-4.33710E+01	0.	0.
11	1.30000E+03	-4.20220E+01	0.	0.
12	1.40000E+03	-4.06630E+01	0.	0.
13	1.50000E+03	-3.92970E+01	0.	0.

JANAF Thermochemical Tables
The Dow Chemical Co.
Midland, Michigan

REPETITION FACTOR A.

CHECK?

? yes

(TERMINAL OUTPUTS FIT)

INPUT REACTION IN FULL ASCII.

? end

(TERMINAL OUTPUTS LIBRARY UPDATE)

READY.

The plotter output is written to file. The printer output is written to the terminal during execution. This allows the operator to examine the fits as they are being produced. If an invalid fitting takes place, the user may take steps appropriate to the situation.

The printed and plotted outputs are shown in Figs. B-1 and B-2. A typical library update is also given.

FENG
13.41.18

PAGE 1.

FENG
76'08 05

$H_2O(g) = H_2(g) + 0.502(g)$

PHASE COEFFICIENT -.500

LIBRARY REFERENCE 1.

BIBLIOGRAPHIC REFERENCE 1.

BIBLIOGRAPHIC REPETITION FACTOR A.

JANAF THERMOCHEMICAL TABLES
THE DOW CHEMICAL CO.
MIDLAND, MICHIGAN

-IDENTIFICATION-

FENG
11.40.27

FENG
76 106 10

PAGE 2.

INPUT DATA

N	T	F
1	3.00000E+02	5.46170E+01
2	4.00000E+02	5.35190E+01
3	5.00000E+02	5.23610E+01
4	6.00000E+02	5.11560E+01
5	7.00000E+02	4.99150E+01
6	8.00000E+02	4.86460E+01
7	9.00000E+02	4.73520E+01
8	1.00000E+03	4.60400E+01
9	1.10000E+03	4.47120E+01
10	1.20000E+03	4.33710E+01
11	1.30000E+03	4.20220E+01
12	1.40000E+03	4.06630E+01
13	1.50000E+03	3.92970E+01

K(T)= 0.

K(F)= 4.61833E+00*(F+F)

-IDENTIFICATION-

FENG
11.40.27

FENG
76.06.10

PAGE 3.

PARAMETERS

I	EXP	P(I)	S(I)
1	L	0.	0.
2	C	2.40204780947E+02	7.96446133155E-02
3	1	-5.04021067121E-02	5.91410435124E-04
4	2	-2.83467362952E-05	1.48177174242E-06
5	3	1.38970049459E-08	1.29764754842E-09
6	4	0.	0.
7	5	0.	0.
8	6	0.	0.
9	7	-1.00873194122E-20	2.44724956178E-21
10	8	1.44618009420E-23	3.93266534997E-24
11	9	-7.59824691596E-27	2.27515178459E-27
12	10	1.41853893161E-30	4.61904513516E-31

- IDENTIFICATION -

FIT
(VAR= 3.10222E-06)

N	T(N) (K)	G(N) (KJ)	Y(N) (KJ)	EPR(N) (KJ)
1	3.00000E+02	2.22907E+02	2.22907E+02	-9.72030E-05
2	4.00000E+02	2.16389E+02	2.16389E+02	4.20482E-04
3	5.00000E+02	2.09619E+02	2.09619E+02	-6.44748E-04
4	6.00000E+02	2.02653E+02	2.02653E+02	2.21067E-04
5	7.00000E+02	1.95536E+02	1.95536E+02	7.58029E-04
6	8.00000E+02	1.88301E+02	1.88300E+02	-1.71678E-03
7	9.00000E+02	1.80963E+02	1.80964E+02	1.87148E-03
8	1.00000E+03	1.73548E+02	1.73548E+02	-5.72220E-04
9	1.10000E+03	1.66067E+02	1.66066E+02	-1.31506E-03
10	1.20000E+03	1.58531E+02	1.58533E+02	1.99523E-03
11	1.30000E+03	1.50962E+02	1.50961E+02	-1.27533E-03
12	1.40000E+03	1.43351E+02	1.43351E+02	4.07934E-04
13	1.50000E+03	1.35711E+02	1.35711E+02	-5.34607E-05

TMIN= 3.00000E+02 K.
TMAX= 1.50000E+03 K.

FREE ENERGY LIBRARY FILE
-IDENTIFICATION-

1. $\text{H}_2\text{O}(\text{g}) = \text{H}_2(\text{g}) + 0.5\text{O}_2(\text{g})$
2. $\text{CO}_2(\text{g}) = \text{C}(\text{s}) + \text{O}_2(\text{g})$
3. $\text{CO}(\text{g}) = \text{C}(\text{s}) + 0.5\text{O}_2(\text{g})$
4. $\text{CH}_4(\text{g}) = \text{C}(\text{s}) + 2\text{H}_2(\text{g})$
5. $\text{OH}(\text{g}) = 0.5\text{O}_2(\text{g}) + 0.5\text{H}_2(\text{g})$
6. $\text{O}(\text{g}) = 0.5\text{O}_2(\text{g})$
7. $\text{H}(\text{g}) = 0.5\text{H}_2(\text{g})$
8. $\text{C}(\text{g}) = \text{C}(\text{s})$
9. $\text{NH}_3(\text{g}) = 0.5\text{N}_2(\text{g}) + 1.5\text{H}_2(\text{g})$
10. $\text{NO}_2(\text{g}) = 0.5\text{N}_2(\text{g}) + \text{O}_2(\text{g})$

FREE ENERGY LIBRARY FILE
-IDENTIFICATION-

$\text{CH}_4(\text{g}) = \text{C}(\text{s}) + 2\text{H}_2(\text{g})$
FE 4 1 A

$\text{CO}_2(\text{g}) = \text{C}(\text{s}) + \text{O}_2(\text{g})$
FE 2 1 A

$\text{CO}(\text{g}) = \text{C}(\text{s}) + 0.5\text{O}_2(\text{g})$
FE 3 1 A

$\text{C}(\text{g}) = \text{C}(\text{s})$
FE 8 1 A

$\text{H}_2\text{O}(\text{g}) = \text{H}_2(\text{g}) + 0.5\text{O}_2(\text{g})$
FE 1 1 A

$\text{H}(\text{g}) = 0.5\text{H}_2(\text{g})$
FE 7 1 A

$\text{NH}_3(\text{g}) = 0.5\text{N}_2(\text{g}) + 1.5\text{H}_2(\text{g})$
FE 9 1 A

$\text{NO}_2(\text{g}) = 0.5\text{N}_2(\text{g}) + \text{O}_2(\text{g})$
FE 10 1 A

$\text{OH}(\text{g}) = 0.5\text{O}_2(\text{g}) + 0.5\text{H}_2(\text{g})$
FE 5 1 A

$\text{O}(\text{g}) = 0.5\text{O}_2(\text{g})$
FE 6 1 A

FENG
16.28.33

FENG
76-08-16

PAGE 3.

FREE ENERGY BIBLIOGRAPHY FILE
-IDENTIFICATION-

1. JANAF THERMOCHEMICAL TABLES
THE DOW CHEMICAL CO.
MIDLAND, MICHIGAN

-IDENTIFICATION-

Fig. B-1.
Typical plotted output from QUIL.

Fig. B-2.
Typical plotted output from QUIL.

APPENDIX C

SAMPLE SESSION WITH SURF

A typical session is outlined wherein a fit is executed with SURF, but is not loaded into the library. The code is put into execution with a procedure file. Once in execution, the code will ask the user for the necessary information to carry out the fit.

The example is a fitting of the model to the iodine data of Iwamoto and Oishi.⁷ In the session illustrated (Fig. C-1) the model is altered by allowing a new parameter that was fixed to be freed in the fit. Input is taken from tapes 7 and 9 (see Sec. XXVIII).

The actual prompts and responses are indicated below. The procedure file SURF puts the code SURFCOD into execution.

```
old,surf
READY.
ascii
```

READY.

-surf

INPUT REACTION IN FULL ASCII.

? I2[s]=I2<g>

DATA ON FILE?

? yes

PARAMETERS ON FILE?

? yes

COEF(I)

L/I	1	2	3	4
1	0.	-4.70000E+00	1.50704E-03	

ROEF(I,L)

1	1.14616E+02	0.	0.	1.00000E+00
2	-8.70133E+02	3.11727E+00	0.	1.00000E+00

SOEF(I,L)

1	1.75664E+03	0.	0.	1.00000E+00
2	0.	0.	2.77738E-03	1.00000E+00
3	0.	0.	5.75581E-03	1.00000E+00

FOEF(I,L)

1	0.	1.20000E+01	0.	1.00000E+00
2	0.	1.00000E+01	0.	1.00000E+00
3	0.	1.00000E+01	0.	1.00000E+00

INPUT PARLIST(COEF,ROEF,SOEF,FOEF).

?

CNULS(I)				
L/I	1	2	3	4
1	T	T	F	

RNULS(I,L)				
1	F	T	T	T
2	F	F	T	T

SNULS(I,L)				
1	F	T	T	T
2	T	T	F	T
3	T	T	F	T

FNULS(I,L)				
1	T	T	T	T
2	T	T	T	T
3	T	T	T	T

INPUT NULLIST(CNULS,RNULS,SNULS,FNULS).
 ? \$NULLIST CNULS(2)=F \$
 ?

CNULS(I)				
L/I	1	2	3	4
1	T	F	F	

RNULS(I,L)				
1	F	T	T	T
2	F	F	T	T

SNULS(I,L)				
1	F	T	T	T
2	T	T	F	T
3	T	T	F	T

FNULS(I,L)				
1	T	T	T	T
2	T	T	T	T
3	T	T	T	T

INPUT NULLIST(CNULS,RNULS,SNULS,FNULS).
?

LIBRARY ENTRY?
? no

I2[s]=I2<g>

CHECK?
? yes

(TERMINAL OUTPUTS FIT)

INPUT REACTION IN FULL ASCII.
? END

READY.

The plotter output is written to file. The printer output is written to the terminal during execution. This allows the operator to examine the fits as they are being produced. If an invalid fitting or a program abort occurs, the user may take whatever steps are appropriate.

The printed and plotted outputs are included in this section.

[illegible]

PAGE 1.

$$12[\varepsilon] = 12 \times 5$$

PHASE COEFFICIENT -1.000

LIBRARY REFERENCE 42.

PREFACE REFERENCE 23.

BIBLIOGRAPHIC REFERENCE 4.

BIBLIOGRAPHIC REPETITION FACTOR R.

K. INAMOTO, AND J. DIEHL
THE BEHAVIOR OF IODINE IN ADSORPTION
AND DESORPTION BY GRAPHITE
J. NUC. EN. TECH., 5(9), 437-446, (1968).

[illegible]

[illegible]

◆ ◆ ◆ ◆ ◆ ◆ ◆ ◆ ◆ ◆

PAGE 2.

INPUT DATA

N	T	C	P
1	3.00000E+02	3.97000E+00	2.70000E-02
2	3.00000E+02	3.66000E+00	1.40000E-02
3	3.00000E+02	3.39000E+00	8.00000E-03
4	3.00000E+02	2.76000E+00	4.00000E-03
5	3.23000E+02	4.17000E+00	1.12000E-01
6	3.23000E+02	3.86000E+00	6.30000E-02
7	3.23000E+02	3.49000E+00	4.10000E-02
8	3.23000E+02	3.26000E+00	2.70000E-02
9	3.23000E+02	2.97000E+00	1.40000E-02
10	3.23000E+02	2.16000E+00	8.00000E-03
11	3.23000E+02	1.31000E+00	4.00000E-03
12	3.36000E+02	3.56000E+00	6.30000E-02
13	3.36000E+02	3.13000E+00	4.10000E-02
14	3.36000E+02	2.83000E+00	2.70000E-02
15	3.36000E+02	1.52000E+00	1.40000E-02
16	3.36000E+02	9.60000E-01	8.00000E-03
17	3.36000E+02	6.60000E-01	4.00000E-03
18	3.48000E+02	3.44000E+00	1.12000E-01
19	3.48000E+02	3.00000E+00	6.30000E-02
20	3.48000E+02	2.51000E+00	4.10000E-02
21	3.48000E+02	1.60000E+00	2.70000E-02
22	3.48000E+02	8.90000E-01	1.40000E-02
23	3.48000E+02	5.60000E-01	8.00000E-03
24	3.48000E+02	3.70000E-01	4.00000E-03
25	3.61000E+02	2.71000E+00	1.12000E-01
26	3.61000E+02	1.60000E+00	6.30000E-02
27	3.61000E+02	1.08000E+00	4.10000E-02
28	3.61000E+02	7.40000E-01	2.70000E-02
29	3.61000E+02	5.60000E-01	1.40000E-02
30	3.61000E+02	3.90000E-01	8.00000E-03
31	3.61000E+02	2.70000E-01	4.00000E-03
32	3.73000E+02	1.32000E+00	1.12000E-01
33	3.73000E+02	8.10000E-01	6.30000E-02
34	3.73000E+02	6.60000E-01	4.10000E-02
35	3.73000E+02	4.70000E-01	2.70000E-02
36	3.73000E+02	3.40000E-01	1.40000E-02
37	3.73000E+02	2.70000E-01	8.00000E-03
38	3.73000E+02	1.90000E-01	4.00000E-03
39	4.73000E+02	1.90000E-01	1.12000E-01
40	4.73000E+02	6.00000E-02	4.00000E-03

```
K(T)= 0.
K(C)= 7.88000E-06(F)
K(P)= 1.00000E+00(F)
K(R)= 6.00600E+00
```

-IDENTIFICATION-

 IUFF
 14.46.20

PAGE 3.

IUFF
 76 08 02

PARAMETERS

CONCENTRATION

	NUL	PAP	SIG
A	T	0.	0.
B	F	-4.81215449468E+00	8.99320710600E-02
C	F	1.50704305223E-03	3.00232036198E-04

RATIO (1)

	NUL	PAP	SIG
A	F	1.14613920576E+02	1.39941618539E+01
B	T	0.	0.
C	T	0.	0.
D	T	1.00000000000E+00	0.

RATIO (2)

	NUL	PAP	SIG
A	F	-8.70133240715E+02	1.72239426592E+02
B	F	3.11726793157E+00	5.22392827896E-01
C	T	0.	0.
D	T	1.00000000000E+00	0.

SLOPE (1)

	NUL	PAP	SIG
A	F	1.75664297086E+03	1.83318131617E+02
B	T	0.	0.
C	T	0.	0.
D	T	1.00000000000E+00	0.

SLOPE (2)

	NUL	PAP	SIG
A	T	0.	0.
B	T	0.	0.
C	F	2.77738182266E-03	2.85040060891E-04
D	T	1.00000000000E+00	0.

 -IDENTIFICATION-

CUFF
14.46.20

PAGE 4.

CUFF
76/09 02

SLOPE (3)

	NUL	PAP	SIG
A	T	0.	0.
E	T	0.	0.
C	F	5.75580705443E-03	3.85612035631E-04
D	T	1.00000000000E+00	0.

MELD (1)

	NUL	PAP	SIG
A	T	0.	0.
E	T	1.20000000000E+01	0.
C	T	0.	0.
D	T	1.00000000000E+00	0.

MELD (2)

	NUL	PAP	SIG
A	T	0.	0.
E	T	1.20000000000E+01	0.
C	T	0.	0.
D	T	1.00000000000E+00	0.

MELD (3)

	NUL	PAP	SIG
A	T	0.	0.
E	T	1.00000000000E+01	0.
C	T	0.	0.
D	T	1.00000000000E+00	0.

-IDENTIFICATION-

 CUFF
 14.46.20

CUFF
 76/08 02

PAGE 5.

ASYMPTOTES

N	TEMP (K)	L	BPEAK POINT	SLOPE	MELD
1	3.00000E+02	1	-4.36004E+00	5.85548E+00	1.20000E+01
		2	-4.74209E+00	9.33215E-01	1.20000E+01
		3	-4.95921E+00	1.72674E+00	1.00000E+01

N	TEMP (K)	L	BPEAK POINT	SLOPE	MELD
2	3.23000E+02	1	-4.32538E+00	5.43852E+00	1.20000E+01
		2	-4.68022E+00	8.97094E-01	1.20000E+01
		3	-5.10359E+00	1.85913E+00	1.00000E+01

N	TEMP (K)	L	BPEAK POINT	SLOPE	MELD
3	3.36000E+02	1	-4.30579E+00	5.22810E+00	1.20000E+01
		2	-4.64692E+00	9.33200E-01	1.20000E+01
		3	-5.17450E+00	1.93395E+00	1.00000E+01

N	TEMP (K)	L	BPEAK POINT	SLOPE	MELD
4	3.48000E+02	1	-4.28770E+00	5.04782E+00	1.20000E+01
		2	-4.61708E+00	9.66529E-01	1.20000E+01
		3	-5.23396E+00	2.00302E+00	1.00000E+01

N	TEMP (K)	L	BPEAK POINT	SLOPE	MELD
5	3.61000E+02	1	-4.26811E+00	4.86605E+00	1.20000E+01
		2	-4.58563E+00	1.00263E+00	1.20000E+01
		3	-5.29256E+00	2.07785E+00	1.00000E+01

N	TEMP (K)	L	BPEAK POINT	SLOPE	MELD
6	3.73000E+02	1	-4.25003E+00	4.70950E+00	1.20000E+01

-IDENTIFICATION-

CURF
14.46.20

CURF
76/08/02

PAGE 6.

N	TEMP (K)	L	BREAK POINT	SLOPE	MELD
6	3.73000E+02	2	-4.55734E+00	1.03596E+00	1.20000E+01
		3	-5.34181E+00	2.14692E+00	1.00000E+01

N	TEMP (K)	L	BREAK POINT	SLOPE	MELD
7	4.73000E+02	1	-4.09932E+00	3.71383E+00	1.20000E+01
		2	-4.34180E+00	1.31370E+00	1.20000E+01
		3	-5.61946E+00	2.72250E+00	1.00000E+01

-IDENTIFICATION-

 SURF
 14.46.20

SURF
 76/08 02

PAGE 7.

FIT
 (VAP= 2.90334E-03)

N	T(N)	LOG C(N)	LOG P(N)	LOG Y(N)	LOG E(N)
1	3.00000E+02	-4.50468E+00	-1.56864E+00	-1.60438E+00	-3.57430E-02
2	3.00000E+02	-4.53999E+00	-1.85287E+00	-1.80831E+00	4.55584E-02
3	3.00000E+02	-4.57327E+00	-2.09691E+00	-2.00129E+00	9.56168E-02
4	3.00000E+02	-4.66256E+00	-2.39794E+00	-2.50619E+00	-1.08253E-01
5	3.23000E+02	-4.48334E+00	-9.50782E-01	-1.01539E+00	-6.46117E-02
6	3.23000E+02	-4.51669E+00	-1.20066E+00	-1.19527E+00	5.38994E-02
7	3.23000E+02	-4.56065E+00	-1.38722E+00	-1.42844E+00	-4.12242E-02
8	3.23000E+02	-4.59026E+00	-1.56864E+00	-1.58209E+00	-1.34503E-02
9	3.23000E+02	-4.63072E+00	-1.85387E+00	-1.77791E+00	7.59593E-02
10	3.23000E+02	-4.76902E+00	-2.09691E+00	-2.15055E+00	-5.36382E-02
11	3.23000E+02	-4.98620E+00	-2.39794E+00	-2.36160E+00	3.63420E-02
12	3.36000E+02	-4.55202E+00	-1.20066E+00	-1.13368E+00	6.69926E-02
13	3.36000E+02	-4.60793E+00	-1.38722E+00	-1.39114E+00	-3.92645E-03
14	3.36000E+02	-4.65169E+00	-1.56864E+00	-1.54711E+00	2.15230E-02
15	3.36000E+02	-4.92163E+00	-1.85387E+00	-1.89690E+00	-4.30243E-02
16	3.36000E+02	-5.12120E+00	-2.09691E+00	-2.09426E+00	2.65204E-03
17	3.36000E+02	-5.28393E+00	-2.39794E+00	-2.34782E+00	5.01326E-02
18	3.48000E+02	-4.56692E+00	-9.50782E-01	-9.81532E-01	-3.07499E-02
19	3.48000E+02	-4.62635E+00	-1.20066E+00	-1.19194E+00	8.71857E-03
20	3.48000E+02	-4.70380E+00	-1.38722E+00	-1.33857E+00	4.86436E-02
21	3.48000E+02	-4.89935E+00	-1.56864E+00	-1.54041E+00	2.82256E-02
22	3.48000E+02	-5.15400E+00	-1.85287E+00	-1.79329E+00	6.05775E-02
23	3.48000E+02	-5.35529E+00	-2.09691E+00	-2.10955E+00	-1.26383E-02
24	3.48000E+02	-5.53527E+00	-2.39794E+00	-2.46743E+00	-6.94944E-02
25	3.61000E+02	-4.67050E+00	-9.50782E-01	-9.71194E-01	-2.04116E-02
26	3.61000E+02	-4.89935E+00	-1.20066E+00	-1.21342E+00	-1.27613E-02
27	3.61000E+02	-5.07005E+00	-1.38722E+00	-1.38486E+00	2.35380E-03
28	3.61000E+02	-5.23424E+00	-1.56864E+00	-1.56004E+00	8.59275E-03
29	3.61000E+02	-5.35529E+00	-1.85387E+00	-1.74791E+00	1.05963E-01
30	3.61000E+02	-5.51241E+00	-2.09691E+00	-2.06479E+00	3.21184E-02
31	3.61000E+02	-5.67211E+00	-2.39794E+00	-2.39634E+00	1.60181E-03
32	3.73000E+02	-4.98290E+00	-9.50782E-01	-1.02821E+00	-7.74262E-02
33	3.73000E+02	-5.19499E+00	-1.20066E+00	-1.24953E+00	-4.88679E-02
34	3.73000E+02	-5.28393E+00	-1.38722E+00	-1.35135E+00	3.58694E-02
35	3.73000E+02	-5.43138E+00	-1.56864E+00	-1.59808E+00	-2.94429E-02
36	3.73000E+02	-5.57199E+00	-1.85387E+00	-1.89444E+00	-4.05691E-02
37	3.73000E+02	-5.67211E+00	-2.09691E+00	-2.10916E+00	-1.22537E-02
38	3.73000E+02	-5.82472E+00	-2.39794E+00	-2.43678E+00	-3.88416E-02
39	4.73000E+02	-5.82472E+00	-9.50782E-01	-9.86008E-01	-3.52260E-02
40	4.73000E+02	-6.32532E+00	-2.39794E+00	-2.34836E+00	4.95834E-02

TMIN= 3.00000E+02 k.
 TMAX= 4.73000E+02 k.

-IDENTIFICATION-

Fig. C-1.
Typical fit from SURF.

REFERENCES

1. R. C. Feber, J. L. Lunsford, and W. A. Stark, Jr., "Application of the Complex Equilibrium Code QUIL to Cesium-Impurity Equilibria in the Primary Coolant of High-Temperature Gas-Cooled Reactors," Los Alamos Scientific Laboratory report LA-NUREG-6373-MS (October 1976).
2. E. E. Osborne, "Smallest Least Squares Solutions of Linear Equations," SIAM J. Numer. Anal. **2**, 300-307 (1965).
3. K. Levenberg, "A Method for the Solution of Certain Non-Linear Problems in Least Squares," Q. Appl. Math. **2**, 164-168 (1944).
4. D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Non-Linear Parameters," SIAM J. Appl. Math. **11**, 431-441 (1963).
5. S. Glasstone, "Free Energy and Chemical Reactions," in *Thermodynamics for Chemists* (van Nostrand, New York, 1947), Chap. XIII, pp. 273-316.
6. *JANAF Thermochemical Tables* (The Dow Co., Midland, MI).
7. K. Iwamoto and J. Oishi, "The Behavior of Iodine in Adsorption and Desorption by Graphite," J. Nucl. Sci. Technol. **5**, 437-446 (1968).