Algorithm in *gem*

The Algorithm used in *gem* for calculating the minimum point of *G*, Gibbs energy of the system.

A unique algorithm has been developed for *gem* in order to solve chemical equilibrium problems in general. This algorithm is mainly based on the gradient projection method with aids of some concepts in linear programming. This algorithm neither needs any estimated initial values other than derived values from the input quantities for reactant mole numbers, temperature and pressure (or volume), nor passes unreasonable states in which mass balance equations will not be satisfied. Therefore, this leads to excellent stability in finding the solution.

1. The problem

Let the system consist of n compounds and m elements. Then, the problem can be described as:

Minimize the Gibbs energy of the system

$$G = \sum_{i=1}^{n} (g_i/RT)X_i + \sum_{i=k}^{n} X_i \ln (X_i P / \sum_{j=K}^{n} X_j) \qquad (1)$$

where x_i: the quantity of the i-th compound (mol)
the 1st to $(k-1)$th compounds are condensed,
and the rest of compounds are gaseous.
P: Pressure (atm)
T: Temperature(K)
R: Gas constant
g_i:Standard Gibbs formation energy of the i-th compound at temperature T
ln : natural logarithm

subject to

$$\begin{aligned} &a_{11}x_1+a_{12}x_2+\ldots+a_{1n}x_n=b_1 \\ &a_{21}x_1+a_{22}x_2+\ldots+a_{2n}x_n=b_2 \\ &\quad\ldots \qquad\ldots \\ &a_{m1}x_1+a_{m2}x_2+\ldots+a_{mn}x_n=b_m \end{aligned} \qquad (2)$$

and

$$x_i \geq 0 \quad i=1, \ldots, n \qquad (3)$$

where

a_{ji}: stoichiometry coefficient of the i-th compound for the j-th element

We will use vector notation, and use primes (') to indicate the transposes of vectors and matrices. Equations (2) can be written as

$$AX'=B \qquad (2)$$

where

$$X = [x_1, x_2, \ldots, x_n]$$

$$A = \begin{bmatrix} A_1 \\ A_2 \\ . \\ A_m \end{bmatrix} = \begin{bmatrix} a_{11}, a_{12}, \ldots, a_{1n} \\ a_{21}, a_{22}, \ldots, a_{2n} \\ \quad\ldots \qquad\ldots \\ a_{m1}, a_{m2}, \ldots, a_{mn} \end{bmatrix}$$

B is constant vector which is given by initial quantities in the system.

The steepest descent vector of G is

$$-\mathrm{grad}\,G = -[\partial G/\partial x_i]$$

We will use this vector and find the minimum point of G under the constraints of (2) and (3). Here, we assume the G is convex in all area defined by (2) and (3).

2. The gradient projection method

If we project the vector $-\mathrm{grad}\,G$ into the space that satisfies (2), the moving point along that direction is always satisfying the mass-balance law. Let $\{A_i\}$ be the space spanned by the vectors $A1$, ..., Am, and S be the space which satisfies (2), then $\{A_i\}$ and S are the orthogonal complement of each other. In other words, if we subtract the components which belong to $\{A_i\}$ from $-\mathrm{grad}\,G$, the resultant vector belongs to S. Hence we first transform the vector series A_i into the orthogonal vectors L_i by the Gram-Schmidt orthogonalization.

Then, we obtain the vector Q as a result of the subtraction of the L_i components from $-\mathrm{grad}\,G$.

3. One stage of calculation for convergence

The initial point of calculation x_0 is given as the quantities of input materials, and we derive the vector Q as above mentioned. The next problem is how far the point should move along that direction. We can derive

$$c = QQ'/QHQ' \quad \text{where } H=[\partial^2 G/\partial x_i \partial x_j] \qquad (4)$$

and adopt $X = X0+cQ$ as the next point based on the ordinary principle of the steepest descent method. This point is, however, either over-going or short-going in general. In order to examine which situation is the case, we make the inner product of $-\mathrm{grad}\,G$ at X and Q. If the inner product is negative, the point is over-going, so that we will adopt some inner point between X and $X0$. If positive, we will go farther.

In the present algorithm, an inter(extra)polating coefficient for each case is given as functions of c, Q and r, which will be described later. By successively applying this method, we finally obtain the point on this direction as the next point where the norm of the inner product is less than an appropriate threshold value.

4. Linear programming

In principle, the gradient projection method is the core of this algorithm. However, we cannot always move along the direction of Q because of non-negative constraints (3). Moreover, if we handle all variables equally at a calculation stage, there would be some problems arisen from the fact that each variable has the value of quite different order.

In order to avoid this problem, our program has employed some concepts utilized in the linear programming. In the linear programming, the movement of point on the convex polyhedron is restricted such as the point should go to the adjacent point only at a calculation stage. That is, the space in which the point is movable in a calculation stage is restricted within one dimensional space including the point.

In our program, the space is being restricted in one dimensional subspace of S. Actually, we make one calculation stage under the condition that $(n-m-1)$ variables out of n variables are regarded to be fixed as constant.

In the next stage, the variable which has the smallest value in the resultant variables is fixed and we take

into account the oldest variable which has been fixed earlier.
Initially, the bigger variable will be served earlier.

For the non-negative constraints, let

$$r=\min\{x_i/|q_i| : q_i<0\} \quad \text{where } [q_i]=Q \qquad (5)$$

and when r is smaller than c in (4), we use this r value instead of c in order to avoid negative variables. In the case of r=0, no calculation is made at the stage and the variable x_i which is 0 and q_i <0 will be fixed in the next stage.

This calculation is iterated until the difference of each variable over n-m successive stages becomes less than 1/10000 of the value.

To confirm that the point satisfies the condition of minimum, the program outputs the norm of the projected grad$G$ (for i : x_i >0) on S. The smaller value indicates that the solution can be regarded as more precise.