# 方程式解法ソフトEQUATRAN
## ——2部グラフの応用

小口 梧郎

### 1 はじめに

EQUATRANは，方程式——線形，非線形および常微分方程式の連立した方程式——を数値的に解くソフトウェアである．EQUATRANでは，方程式をほぼそのまま書き並べるだけで簡便に数値的な結果が得られるため，数学モデルを用いた解析やシミュレーションを行う手段として，BASICやFORTRANの代替として，技術者や研究者を中心に広く利用されつつある．

EQUATRANはもともと化学プロセスのシミュレーションのためのツールとして開発された．プロセスの開発段階のシミュレーションでは，しばしばその数学モデルを作り直したり，入力と出力の関係を入れ換えて計算することが必要になる．このため，汎用のシミュレータでは扱いにくく，従来はその場限りのプログラムを作って対応していた．方程式をそのまま書き，入力と出力とを指定するだけで自動的に解法の手続きが生成できれば大幅な能率向上が可能である．しかしこのためには，方程式を解くという数学的な機能に加えて，様々な対象を容易にかつ柔軟にモデル化する機能が必要である．方程式を数値的に解くためのソフトウェアは他にもいくつかあるが，EQUATRANは実際的な問題をモデル化する機能の豊富なこと，またそれらの機能を自由に組み合わせて使える柔軟性において特に優れている．

EQUATRANのこのような特長は，ここで採用している2部グラフを用いた数学モデルの表現法に負うところが大きい．そこで本稿では，EQUATRANにおける2部グラフの応用を中心に解説することとした．2部グラフはグラフ理論において，「頂点が互いに辺を共有しない2つの集合に分

[筆者紹介]

おぐち ごろう．1966年三井東圧化学(株)に入社．1968年より同社システム部にてプロセス制御用ソフトウェア，プロセスシミュレーションプログラムの開発に従事．1982年日本燃料技術開発(株)に出向，燃料電池発電システムの開発に従事．1985年三井東圧化学(株)システム部に復帰，EQUATRANなど技術計算ソフトウェアの開発に従事，現在に至る．知識表現の研究に興味を持つ．日本機械学会，情報処理学会会員．

けられるグラフ」として定義される．一般のグラフに較べて利用される機会が少ないようであるが，方程式ばかりでなく他の方面においてもモデルを表現する手段として優れていると思われる．本稿を通じてその有用性をご理解いただければ幸いである．

EQUATRAN は，数学モデルを方程式などで記述するための文法，文法に従って記述されたテキストを解釈し 2 部グラフで表現したのち計算の手順を自動生成するコンパイラ，数値計算を実行する数値計算パッケージ，計算結果をグラフ(図)化するグラフ作成ツールなどから構成されている．本稿ではこれらのうちコンパイラの機能を中心に解説することになる．したがって，文法やグラフ化の機能などについては成書等を参照していただきたい[1][2]．なお，現在の EQUATRAN には，大型計算機やエンジニアリングワークステーションのための，EQUATRAN-G とパソコン用で EQUATRAN-G のサブセットにあたる EQUATRAN-M の 2 つの版があるが，ここでは EQUATRAN-G を対象として解説する．

## 2 EQUATRAN の数学モデル

図 1 は簡単な連立方程式を解く問題を EQUATRAN の文法に従って記述したもので，これをソーステキストと呼んでいる．このように，ソーステキストは方程式のほかに入出力と計算方法を指示する文を順不同に書き並べたものである．これを EQUATRAN に入力すれば図 2 のように数値による解が得られる．実際的な問題をモデル化するには，通常の代数方程式と常微分方程式だけでは不十分なことが多い．このため，EQUATRAN には次のような要素が含まれている．

- 指数関数，三角関数などの超越関数
- 整数値化，最大，最小などの特殊な関数
- 論理演算を含む式
- 条件付きの式
- 数値表の形で定義された関数
- ユーザー関数
- 最適化(最大値，最小値の探索)

```
 1:      /* 例題 */
 2:
 3:          d = 3
 4:          4*y + z = c^2
 5:    eq1: exp(z) + d - y = x
 6:          p = loge(y) + 0.5
 7:          x' + 3*y = 0
 8:
 9:      INPUT c
10:      RESET y BY eq1
11:      INTEGRAL t[0,1] STEP 0.01
12:      TREND p,x,y,z STEP 0.1
```

図 1 ソーステキスト(例題)

```
[   入力データ   ]
c                =  2
x                #  10
```

| t | 1:p | 2:x | 3:y | 4:z |
|---|---|---|---|---|
| 0 | -0.2003844 | 10.000000 | 0.4963945 | 2.014422 |
| 0.1000000 | -0.1906110 | 9.850354 | 0.5012697 | 1.994921 |
| 0.2000000 | -0.1806488 | 9.699224 | 0.5062884 | 1.974846 |
| 0.3000000 | -0.1704884 | 9.546566 | 0.5114587 | 1.954165 |
| 0.4000000 | -0.1601199 | 9.392333 | 0.5167894 | 1.932843 |
| 0.5000000 | -0.1495324 | 9.236475 | 0.5222900 | 1.910840 |
| 0.6000000 | -0.1387141 | 9.078941 | 0.5279709 | 1.888116 |
| 0.7000000 | -0.1276522 | 8.919673 | 0.5338437 | 1.864625 |
| 0.8000000 | -0.1163329 | 8.758614 | 0.5399207 | 1.840317 |
| 0.9000000 | -0.1047410 | 8.595699 | 0.5462159 | 1.815137 |
| 1.0000000 | -0.09285999 | 8.430861 | 0.5527442 | 1.789023 |

**図2　例題の計算結果**

・不連続現象

ここで，条件付きの式とは，変数の値の範囲などによって適用される式が異なるもので，たとえば管内の流体の圧力損失を求めるとき，レイノルズ数によって摩擦係数の算式が異なる場合のモデルがこれにあたる．またユーザー関数は，ユーザーがFORTRANのサブプログラムなどの形で用意した関数である．ユーザー関数はその中身がブラックボックスであること，また一般にユーザー関数では，出力変数が複数あり得る点が他の関数や方程式と異なる．不連続現象は常微分方程式を数値積分によって解く過程で変数の値を不連続的に変更する機能であり，物体の衝突に代表される不連続的な現象をシミュレーションするときに有効な機能である．

これらの要素が混在する問題を矛盾なく扱うには，これらを統一的に表現しかつ変形などの処理が可能な手段が必要である．

## 3　グラフ表現

方程式などを記述したソーステキストはその文法に従って解釈され，数学モデル(方程式など)はグラフとして内部的に表現される．このグラフによる表現法はEQUATRANの大きな特徴であり，計算手順の生成の処理は全てこのグラフに基づいて行われるなど重要な役割を果たしている．

（1） 方程式のグラフ

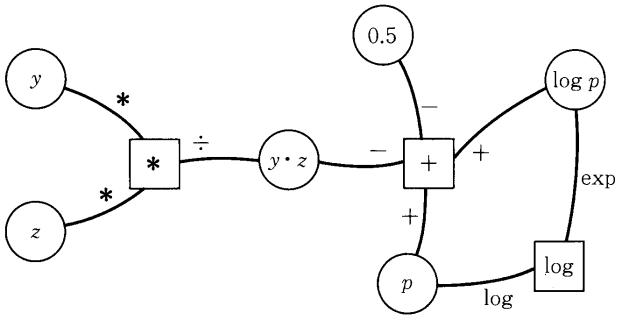



**図3　$y \cdot z = \log(p) + p - 0.5$ のグラフ**

方程式は(一般の方程式以外の関係式も含めて以後単に方程式と呼ぶ),「変数」を一方の頂点とし,「演算」を他方の頂点とする2部グラフによって表される．ここで言う変数とは本来の変数のほかに，定数や演算の結果として現れる中間的な変数を含むものである．たとえば,

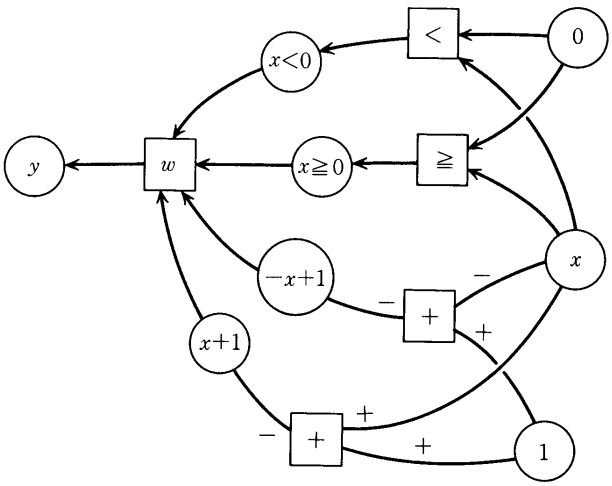
$$y \cdot z = \log(p) + p - 0.5 \qquad (1)$$

においては，$y, z, p, 0.5, y \cdot z, \log(p), \log(p)+p-0.5$ が変数の頂点として表される．一方の演算はこれら変数間の演算や関数関係である．図3に(1)の方程式に対応するグラフ表現を示す．変数の頂点は○，演算の頂点は□によって表している．頂点の間を結ぶ辺には各変数の式に対する関係を示す記号が付されている．この記号の意味は次のように理解するとよい．(1)式を中間変数 $u$ を用いて演算の種類ごとに分解して書き直す．この際，乗算と除算，加算と減算は同種の演算であるとし，1つの式に纏めるとともに乗算あるいは加算記号のみで表せるように左辺と右辺を定める．また関数は常に右辺の変数についてとるように書き表す．

$$\left.\begin{aligned} u_1 &= y \cdot z \\ u_2 &= \log(p) \\ u_1 + 0.5 &= u_2 + p \end{aligned}\right\} \qquad (2)$$

このような一種の正規化をした場合の(2)式の各式が2部グラフにおける演算の頂点であり，各辺に付された記号は対応する変数が式の左辺に属するか右辺に属するかを示しているわけである．

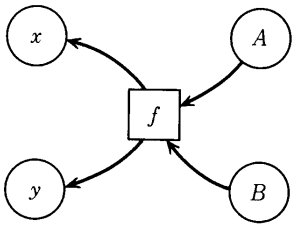
図3のグラフでは辺に方向がないが，辺の方向は計算の方向を表すものと定義されている．すなわち，演算から変数へ向かう辺は変数がその演算によって




(1) 条件付きの式

$$y = \begin{cases} -x+1 & x<0 \text{ のとき} \\ x+1 & x \geqq 0 \text{ のとき} \end{cases}$$

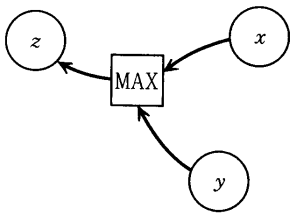
のグラフ




(2) ユーザー関数

$(x, y) = f(A, B)$

のグラフ




(3) 最大値関数

$z = \mathrm{MAX}(x, y)$

のグラフ

**図4 各種のグラフ**

計算される出力であることを，変数から演算に向かう辺はそれらの変数が演算の入力であることを表す．一般の方程式では最初は辺の方向は与えられておらず，後に述べる計算の順序を決める過程でその方向が与えられる．

一方，特殊な関数，ユーザー関数，条件付きの式，論理演算など計算の方向がもともと制限されている演算では，最初から辺の方向が与えられている．図4にいくつかの例を示す．

個々の方程式について以上のようなグラフ表現を行えば，連立方程式は全体として1つの大きな2部グラフとして表現されることになる．実際の例は後に計算手順の生成法とともに示すが，その前に，数値積分，最適化計算あるいは非線型方程式のための繰り返し計算についての指定方法とグラフ表現について述べておく．

（2） 常微分方程式と数値積分のグラフ

EQUATRAN では常微分方程式も他の方程式と同様に取り扱うことができる．表記法は微分記号「′」を用いて，たとえば，

$$\frac{dx}{dt} + a \cdot x = \sin(t) \qquad (3)$$

はソーステキストに，

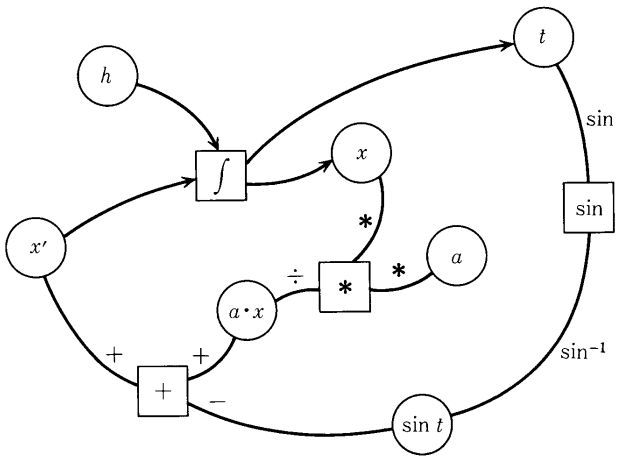
x′+a*x = sin(t)

のように書けばよい．微分の独立変数は別に INTEGRAL 文と呼ばれる宣言文によって，積分範囲などとともに指定する．たとえば，

INTEGRAL t[0,100] STEP h BY RKF

は，$t$ を独立変数とし 0 から 100 の範囲をきざみ $h$ で，固定きざみのルンゲクッタ法により積分すべきことを指定している．

常微分方程式のグラフは，微分項の $x'$ を1つの変数として扱って，一般の方程式と同様に作ればよい．また数値積分の操作は




x′+ax = sin(t)
INTEGRAL t[0,100] STEP h

**図5 微分方程式と積分のグラフ**(陽公式)

1) 積分公式を用いて，次の計算点における被積分変数(この場合 $x$)を計算し

2) この値で被積分変数の値を置き換えて独立変数を進める

ことであるから，この操作を演算の一種と考えて1つの頂点で表すことができる．(3)式の例は図5のグラフになる．図中の $\int$ が積分操作の演算の頂点である．このように微分方程式だけでなく積分操作も演算の一種としてグラフに加えることによって，後に説明する計算順序の生成のためのグラフ処理の際，積分操作のおよぶ範囲(数値積分の計算ループの内側で行うべき演算)が厳密に判別できる．

図5は積分公式として陽公式を用いる場合である．陰公式の場合には新しい計算点における被積分変数の計算にその点での微分項が必要になるが，図5ではこの「陰」の関係が明示的に表されていない(なぜ明示的に表されていることが望ましいかは後に述べる)．そこで，積分公式を方程式として2部グラフに取り入れる．たとえば陰公式の1つである後方オイラー法では，

$$x = x_{-1} + x' \cdot h$$

を加えて図6のグラフが得られる．$x_{-1}$ は1きざみ前の被積分変数を表す．この場合は積分操作の演算 $\int$ は単に変数を置き換えて独立変数を進める機能を表している．

微分方程式において，微分項 $x'$ は $x$ の関数(一般的には非線型の関数)と考えられるので，陰公式とともに非線型の連立方程式を構成する．このため，通常 $x$ を求めるには繰り返し計算が必要になる．陰公式自体を明示的にグラフに含めるのは，この連立方程式を他の数学モデルの方程式と同じレベルで扱えるようにし，全体として合理的な計算順序を生成するためである．

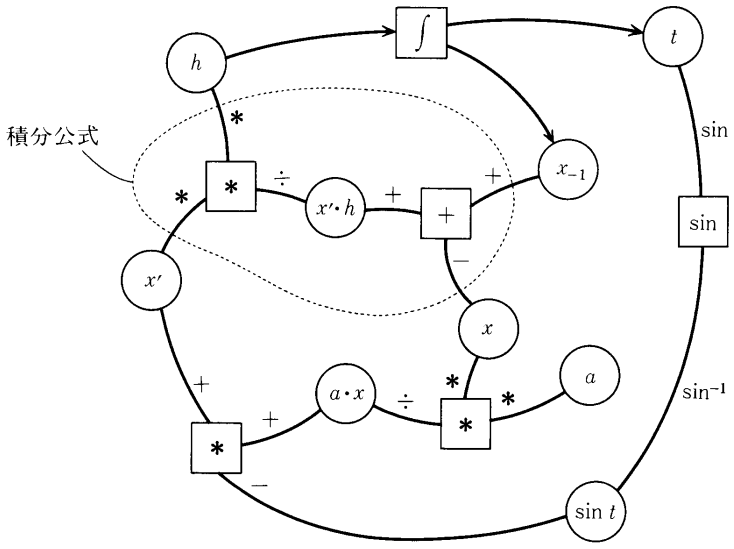
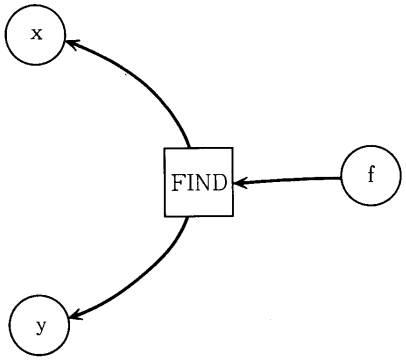
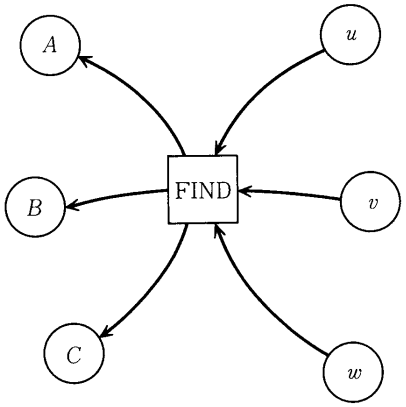



**図6 微分方程式と積分のグラフ**(後方オイラー法)




FIND (x, y) MAXMIZE f

**図7 最適化のグラフ**




FIND (A, B, C) LEAST (u, v, w)

**図8 最小2乗法のグラフ**

(3) 最適化計算とそのグラフ

最適化計算はある変数(評価変数)の値を最大ないし最小にするように別の変数(独立変数)の値を探索法で求める機能である．最適化計算はたとえば次のようなFIND文と呼ばれる宣言文を書いて指定する．

FIND(X, Y) MAXIMIZE f

ここでX, Yが独立変数，fが評価変数である．EQUATRANでは，多変数の最適化計算手法としてコンプレックス法を用いている．この手法では計算された評価変数の値によって新たな探索点(X, Y)を決定する．したがって最適化計算はグラフ上では図7のように表すことができる．

最適化計算の特殊な場合として最小2乗法がある．同様にFIND文によって，

FIND(A, B, C) LEST(u, v, w)

のように指定する．この例では，残差変数 $u, v, w$ の2乗和を最小にする独立変数 $A, B, C$ を求める．最小2乗法の指定をすると手法としてマルカート法が採用されるので，一般の最適化計算として扱うより高速に収束する．マルカート法では残差変数の独立変数によるヤコビ行列が必要になるが，EQUATRANではこれを数値微分によって求めるので，最小2乗法の計算は機能としては残差変数から新たな独立変数を計算する演算と考えることができる．したがって最小2乗法のグラフは図8のようになる．

(4) 繰り返し計算の指定

非線型の方程式あるいはその連立方程式は繰り返し計算によって解く．$n$ 元の非線型連立方程式において，その $n$ 個の変数から適当な $m$ 個を選んでその値を既知と仮定したとき，残りの $n-m$ 個の変数が $n-m$ 個の方程式を使って順次計算できるとすれば，使われなかった $m$ 個の方程式が成立するように選ばれた $m$ 個の変数の値を探せば元の連立方程式が解かれたことになる．選ばれた $m$ 個の変数を独立変数，計算に使わなかった $m$ 個の方程式をチェック

の式と呼ぶことにすると，独立変数の選び方とチェックの式の選び方にはいずれも任意性があるが，これらを決めてしまえば計算の順序は一意に決まる．

独立変数の数 $m$ は少ない方が一般には繰り返し計算の計算量も少なく収束しやすいことが期待されるが，実際には個々の方程式の性質に大きく左右されるためそうならないことが多い．また，工学的な問題を扱う場合には，モデル化の対象とする現象から，どのような計算法を取るのがよいかが見当が付いていることが多い．EQUATRAN では $m$ をなるべく小さくするようにこの $m$ 個の変数と式の組合せを自動的に選びだすアルゴリズムを持っているが，一方，ユーザーがこの組合せを積極的に指定することによって繰り返し計算の方法をコントロールできるようにしている．

繰り返し計算の指定は次のような RESET 文によって行う．

```
RESET X BY eq1,
      Y BY eq2
```

この例では独立変数として X と Y，チェックの式として eq1 と eq2 を指定している．ここで eq1, eq2 は方程式に付けられたラベルである．繰り返し計算の手法としてはニュートンラフソン法が用いられ，必要なヤコビ行列は数値微分によって計算される．繰り返し計算の指定は，最適化計算の場合と異なり方程式系の自由度に影響を与えないので，グラフには表現されない．

## 4 計算順序の生成

次に 2 部グラフを利用して表現された連立方程式の計算順序を生成する手順を説明する．例として図 1 のソーステキストで表現された問題を採り上げる．この例は小さいながら非線型方程式と常微分方程式が連立した形になっており，

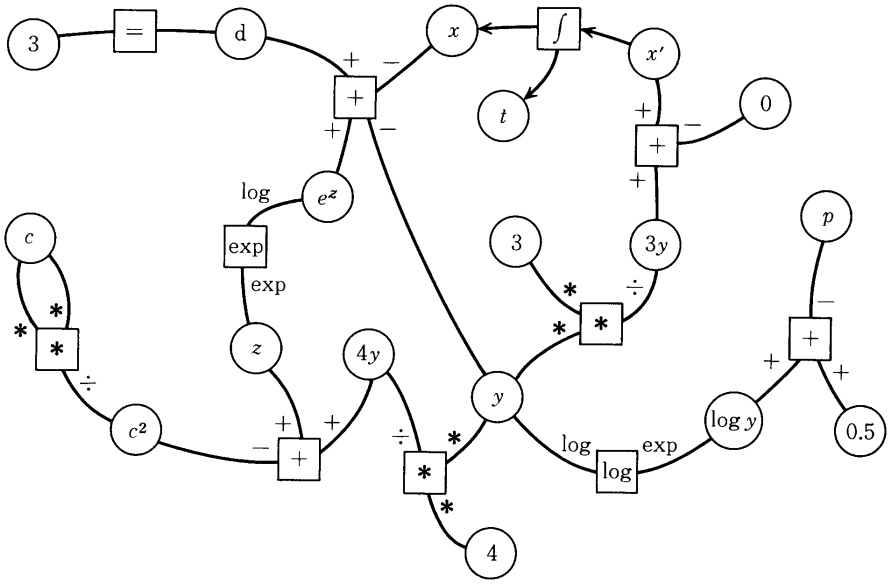



**図 9 図 1 の例題の全体のグラフ**

計算の順序としては数値積分法のループの中で非線型方程式を繰り返し計算で解かなければならない．なお，この例は[3]で採り上げられている例題にユーザーによる繰り返し計算の指定を加えたものになっている．

（1） 辺の方向付けと可解性のチェック

図9がこの例題の2部グラフである．最初のステップはこの2部グラフの中の全ての無向辺に方向を与えることである．辺の方向は次の条件を満たすように決める．

1) 定数と入力変数の頂点には外向きの辺だけが許される．ここで入力変数とは，計算の実行時にユーザーが値をインプットする変数で，例題のcがこれにあたる．
2) その他の変数の頂点には1つかつ1つだけの内向きの辺が存在する．
3) 演算の頂点には1つかつ1つだけの外向きの辺が存在する．

これらの条件を満たすように方向付けを行うことは，演算の頂点と未知変数の頂点とをそれぞれ一対一に対応付けを行うこと，すなわち2部グラフにおけるマッチングに相当する．マッチングが成功するか否か，つまり全ての頂点の対応付けが実現できるか否かは与えられた連立方程式が解のある形をしているか否か(構造的に可解か否か)を示している．未知変数と式の数が一致していない場合や，全体として数が同じでも局部的な過不足がある場合がマッチングを行うことによって判別される．

EQUATRANにおけるマッチングのアルゴリズムは概略次のようである．

1) 上の3つの条件を可能なかぎり満たすように全ての無向辺の方向を仮に決定する．
2) 入力辺が不足している頂点があれば，この頂点から辺の方向に沿って下流を探索し，入力辺が過剰となっている頂点を探す．見つかれば不足の頂点から過剰の頂点に到る1つの順路上の辺の方向を全て逆転する．
3) 2)の操作を可能なかぎり繰り返し，過不足の頂点が無くなればマッチングは成功し，1つでも残ればマッチングは失敗する．

この辺の方向を逆転する操作においては，2で述べた特殊な関数のようにあらかじめ方向が与えられている辺の扱いが問題になる．論理演算や整数値化の関数のように逆方向の計算が定義できないものについては辺の方向の逆転は禁止される．また，ユーザーの関数や絶対値の関数などでは逆関数は用意されていないが繰り返し計算を前提すれば逆方向の計算も可能と考えてよいので辺の方向の逆転が許される．

なお，一般的な2部グラフのマッチングのためのアルゴリズムはたとえば[4]などにある．

（2） サイクリックブロックの判別

方向付けを終わったグラフを図10に示す．グラフの方向付けが無事終了す

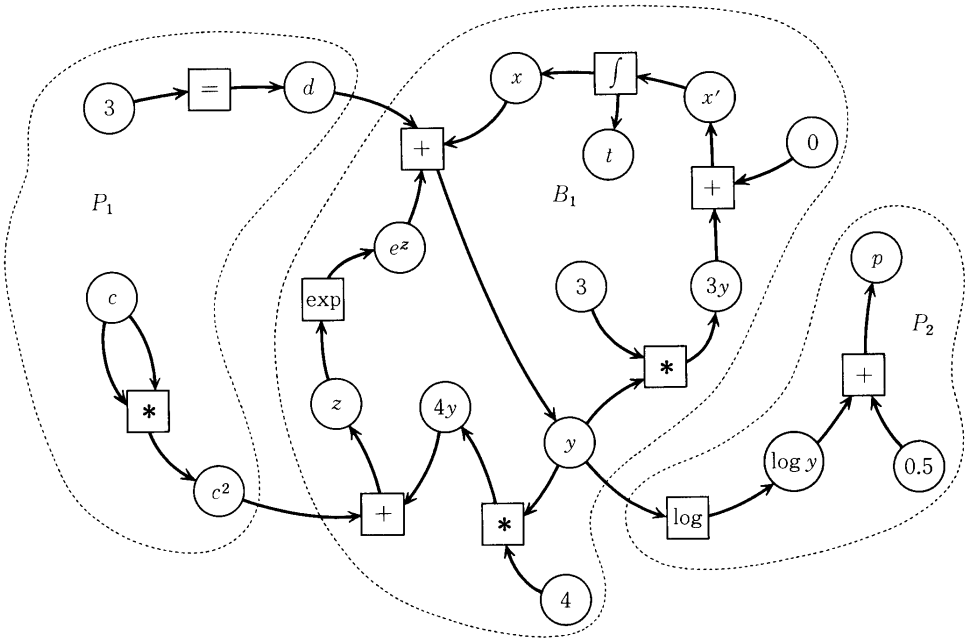



**図 10　方向付けを終わったグラフ**

れば，この方向に沿って上流から順に計算を進めればよいのであるが，図の $B_1$ の部分のように辺の方向に沿ってループが構成されている部分は直ちに計算順序を決めることはできない．このような部分をサイクリックブロックと呼んでいる．サイクリックブロックは次の場合に生じる．

1)　線型あるいは非線型の方程式で連立して解くべき部分

2)　常微分方程式の数値積分のループ内の計算

3)　最適化計算のループ内の計算

1つのグラフ内には複数のサイクリックブロックが生じうるが，各頂点が同一のサイクリックブロックに属するか否かは互いに往復可能な順路が存在するか否かによって判別される．

サイクリックブロックの判別を行った後，この内部の計算順序の決定をとりあえず保留しておけば，グラフの全体はサイクリックブロック($B_i$)とループを含まない部分である非サイクリックブロック($P_i$)との順序付きの列に置き換えることができ，各ブロックの計算順序が決められる．図 10 の場合は

$$(P_1) \rightarrow (B_1) \rightarrow (P_2)$$

の順に計算すればよいことが分かる．非サイクリックブロックの中の計算順序は上流から順次決めていけばよいので，後はサイクリックブロック内の計算手順を決めればよいことになる．

このようにサイクリックブロックと非サイクリックブロックとに分解することによって，繰り返して実行が必要となるサイクリックブロック内の演算を最少にすることができる．なおこの分解はグラフ理論では有向グラフの強連結成分への分解と呼ばれている．

（3） 線型連立方程式のサイクリックブロック

次の2つの条件を満たせばそのサイクリックブロックは線型の連立方程式である．

1) ブロック内の演算の頂点はすべて加減算か乗除算である．

2) 乗除算の頂点と辺を共有する変数の頂点はブロック内には2つだけ存在し，その2つの辺には異なる記号(*と ÷)が付されている．

線型の連立方程式はそのまま線型代数の手法で解けばよい．なお，この連立方程式の係数行列はスパースであることが多いので，そのための解法を用意しておく必要がある[5]．

（4） サイクリックブロックの分解

残ったサイクリックブロックに対しては以下の操作を順次適用する．

1) ブロック内に積分操作の演算の頂点があればこれを取り除き，このブロックの周りに数値積分のための計算ループを設定する．

2) ブロック内に最適化計算の演算の頂点があればこれを取り除き，このブロックの周りに最適化計算のための計算ループを設定する．

3) ブロック内に繰り返し計算の指定を受けている変数と演算の頂点があれば，変数の頂点から演算の頂点にいたる1つの順路上の辺の方向を全て逆転したのちこの2つの頂点を取り除き，このブロックの周りに繰り返し計算のための計算ループを設定する．

4) ブロック内から繰り返し計算のための独立変数の頂点とチェックの式を一組選定し，3)と同様な操作を行う．

適用できる操作が1つでもあればサイクリックブロックは分解される．分解

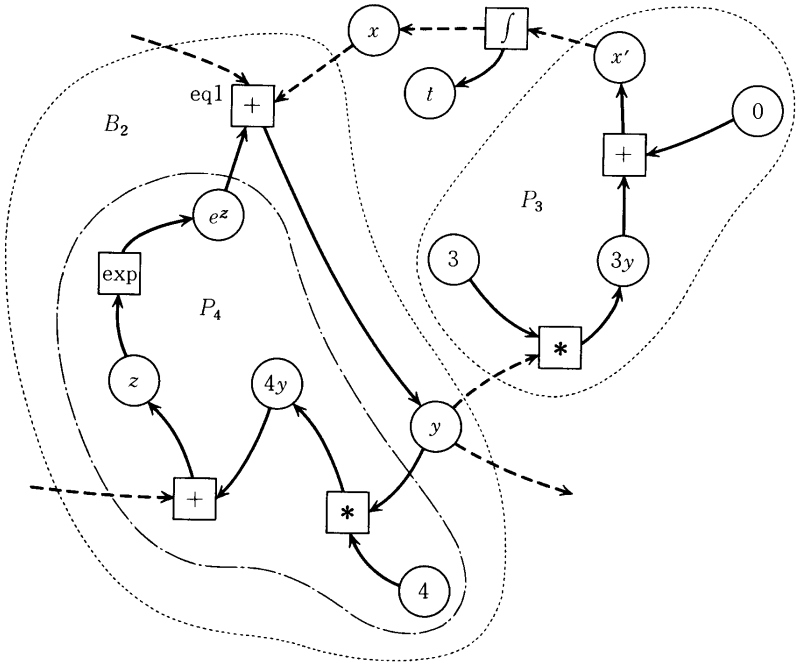



**図11 サイクリックブロック $B_1$ の分解**

後のグラフに対して(2)のサイクリックブロックの判別以下の手順をサイクリックブロックが消滅するまで再帰的に繰り返すことによって全ての演算の計算順序が決定される．

例題のサイクリックブロック $B_1$ に対してはまず積分操作の頂点によって分解が行われ，分解されたグラフから新たにサイクリックブロック $B_2$ が判別される．この段階では全体の計算順序は次のようになっている(図 11 参照)．

$$(P_1)\rightarrow(B_2)\rightarrow(P_3)\rightarrow(P_2)$$

数値積分のための計算ループ

さらにブロック $B_2$ に対して繰り返し計算による分解が行われ，最終的には次のように計算順序が定まる．

繰り返し計算のための計算ループ

$$(P_1)\rightarrow\ \rightarrow(P_4)\rightarrow(P_3)\rightarrow(P_2)$$

数値積分のための計算ループ

## 5 数値微分とグラフ上の微分

前にも触れたように，EQUATRAN ではニュートンラフソン法のためのヤコビ行列を得るために数値微分を用いているが，解析的に微分する方法も考えられる．EQUATRAN への応用という面から両者を比較すると，数値微分の利点としては，

1) 数値微分では偏微係数を計算するとき，関数値の計算と同じ手順を繰り返し用いればよいので，計算コードの量が少なくてすむ．特にパーソナルコンピュータのように使用可能なメモリーが制限される処理系を用いる場合には，この点は重要となる．

2) 関数関係が通常の方程式ばかりでなく，ユーザーの作成したプログラムとして与えられた場合にも対応可能である．

一方欠点としては，

3) 数値微分のための摂動幅が適切でないと正しい偏微係数が得られない．

4) 独立変数の数が多い場合には，解析的な方法の方が一般に高速である．すなわち数値微分では，特に工夫をしないと，独立変数の数に等しい回数だけ関数の計算が必要となるのに対して，[6]によれば，解析的な方法では独立変数の数によらず関数計算の定数倍の計算量でヤコビ行列を得ることができる．

3)に関しては，EQUATRAN では適切な摂動幅で数値微分が行えるようくつかの工夫をしている．たとえば，独立変数についてはその変域をユーザー

が指定することができるが，摂動幅をこの変域の幅に連動させることによって不適切な値を取らないようにしている．また，ニュートンラフソン法の繰り返しの過程で，十分微小な独立変数の移動によっても解の改善が得られないなど，ヤコビ行列が正しく計算されていないと判断される場合には，自動的に摂動幅の修正を行う．このような処理によって，実用上は数値微分に起因するトラブルを避けることができる．

次に，EQUATRAN には採用されていないが，2部グラフを利用して解析的な微分に基づくニュートンラフソン法の手順を生成する方法を紹介する．

次の形の方程式を考えてみる．

$$\left.\begin{aligned} f &= f(x, y, u) \\ g &= g(x, y, u) \\ u &= u(x, y) \end{aligned}\right\} \tag{5}$$

ここで，$x$ と $y$ とを独立変数として，$f=0, g=0$ を満たす解を求めるものとする．一般のニュートンラフソン法の手順では，(5)式から中間変数の $u$ を消去したのち，仮定した$(x, y)$における，ヤコビ行列

$$J = \begin{pmatrix} \partial f/\partial x & \partial f/\partial y \\ \partial g/\partial x & \partial g/\partial y \end{pmatrix} \tag{6}$$

を計算し，$x$ と $y$ の修正量を

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = J^{-1} \begin{pmatrix} \Delta f \\ \Delta g \end{pmatrix} \tag{7}$$

で計算する．$\Delta f$ と $\Delta g$ は $f$ と $g$ に対する必要な修正量であり，すなわち $\Delta f = -f, \Delta g = -g$ である．

このニュートンラフソン法の操作は次のように考えてもよい．まず，(5)の各式についてそのままの全微分形を求める．

$$\left.\begin{aligned} \mathrm{d}f &= \frac{\partial f}{\partial x}\mathrm{d}x + \frac{\partial f}{\partial y}\mathrm{d}y + \frac{\partial f}{\partial u}\mathrm{d}u \\ \mathrm{d}g &= \frac{\partial g}{\partial x}\mathrm{d}x + \frac{\partial g}{\partial y}\mathrm{d}y + \frac{\partial g}{\partial u}\mathrm{d}u \\ \mathrm{d}u &= \frac{\partial u}{\partial x}\mathrm{d}x + \frac{\partial u}{\partial y}\mathrm{d}y \end{aligned}\right\} \tag{8}$$

ここで，$\mathrm{d}f$ および $\mathrm{d}g$ を $f$ および $g$ についての必要な修正量と考えれば，(8)式における $\mathrm{d}x$ と $\mathrm{d}y$ はそれに対応する $x$ と $y$ の修正量にほかならない．したがって次の関係

$$\left.\begin{aligned} f &= -\mathrm{d}f \\ g &= -\mathrm{d}g \end{aligned}\right\} \tag{9}$$

を追加すれば，ニュートンラフソン法の1回の計算は，(5)(8)(9)の連立方程式を $x$ と $y$ を既知として解いて $\mathrm{d}x$ と $\mathrm{d}y$ を求めることと同等である．この連立方程式にはたかだか線型のループブロックしか含まれないから，3項の操作

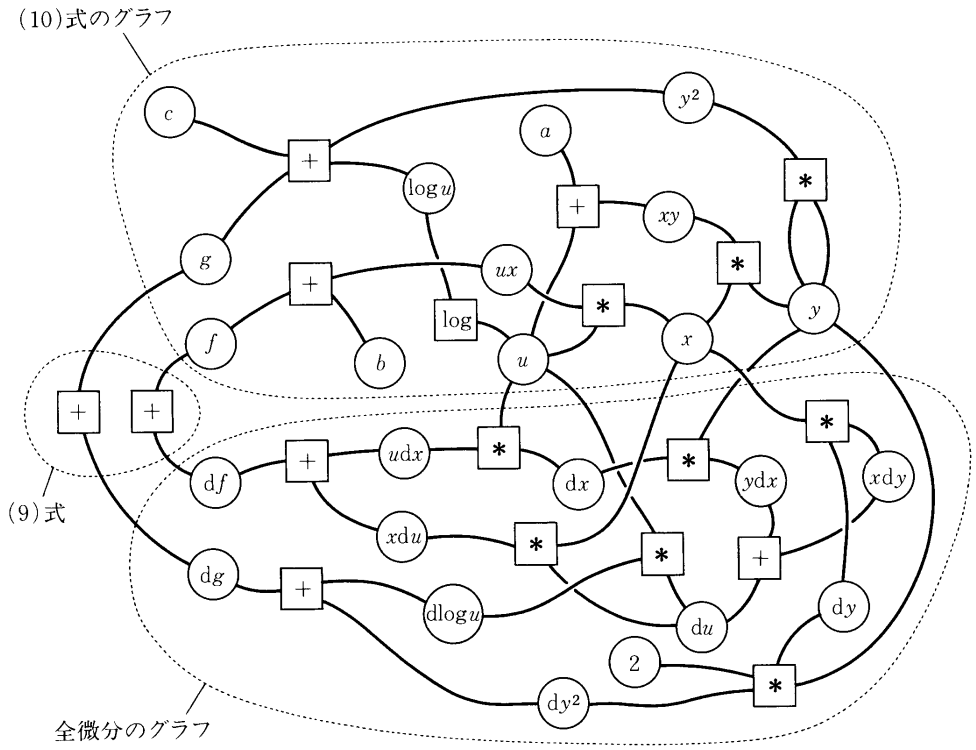



**図 12　全微分の式と(9)式を含むグラフ**

で繰り返し計算を含まない解法の手順を作ることができる．次の連立方程式を例にして実際に全微分のグラフを求めてみる．

$$\left.\begin{aligned} \mathrm{f} &= u\cdot x+b \\ g &= \log(u)-y^2+c \\ u &= x\cdot y+a \end{aligned}\right\} \qquad (10)$$

2 部グラフでは元のグラフの各演算の頂点について全微分を機械的に行えば全体の全微分のグラフが簡単に得られる．定数についての全微分が 0 であることも利用して，元の方程式(10)と全微分の式および(9)式を併せたグラフが図 12 のように得られる．

全微分を含むグラフの規模は，作成の手順からも明らかなように元の方程式のグラフのたかだか数倍にすぎないと考えてよい．実際この例では元のグラフの辺の数が 21 であるのに対し，全微分を含むグラフの辺の数は 54 である．d$x$ と d$y$ を求めるために元数の多い線型方程式を解くことが必要になるが，この方程式はきわめてスパースなので，計算の量はヤコビ行列を用いる方法と同程度で実現できると期待される．

なお，この全微分のグラフは，繰り返し計算の方法，すなわち独立変数とチェックする式の選び方によらず同じである．したがってこのグラフを用いれば，繰り返しの途中で動的に繰り返し計算の方法を変更することができる．

## 6 おわりに

方程式を演算の頂点と変数の頂点とからなる2部グラフを用いて表現することによってEQUATRANの機能が実現されていることを示した．2部グラフを用いることの意味は，辺の向きが変更可能であることと密接に関係がある．それぞれの方程式においてその計算の向きが固定されており，すなわち式の変形を行わないことを前提とすれば，変数の頂点だけあるいは演算の頂点だけからなる有向グラフで十分である．変数だけを用いたグラフは「計算グラフ」と呼ばれ，計算過程を表現する手段として利用されている[6]．

化学プロセスのシミュレーションの分野では古くからグラフが利用されている．考えてみれば，化学プラントはグラフそのものである．タンクや反応器は頂点であり，これらを結ぶパイプは辺，パイプの中の流体の流れの方向は辺の向きである．実際，プロセスのシミュレーションではこのような有向グラフを用いて各装置の計算順序を決定することが行われる．しかし，現実のプロセスでの現象や情報の伝わる向きはいつも流れの方向に一致しているわけではない．下流のバルブを絞れば上流の流量や圧力が変化する．反応器の温度を維持するために上流の加熱器の燃料を制御する必要もある．このような柔軟性のあるプロセスのモデルは，装置での計算を一方の頂点とし，流量などのプロセス変数をもう一方の頂点とする，無向辺を含んだ2部グラフで表現することができる[7]．ほかにもこのような2部グラフの応用分野は少なくないと思われる．

[参考文献]

[1] 三井東圧EQM研究会，パソコンのための方程式解法ソフトEQUATRAN-M入門，省エネルギーセンター，東京，1987.
[2] 畠山淳一，パソコンソフト活用事例，電機書院，東京，1990.
[3] 小口梧郎，横山克己，化学工業における数値計算(6)EQUATRAN-Mの機能と構造，information, Jan., 1987.
[4] Papadimitriou, C. H., and Steiglitz, K., Combinatorial Optimization——Algorithms and Complexity, Prentice-Hall, 1982.
[5] Tewarson, R. P., Sparse Matrices, Academic Press, 1973.
[6] 伊理正夫，土谷隆，星守，偏導関数計算と丸め誤差推定の自動化の大規模非線型方程式への応用，情報処理，26, 11(1985), 1411-1420.
[7] Oguchi, G., Kano, O., and Mitsunaga, M., Process Flowsheet Simulation by PSX, Preprint of International Congress, Contribution of Computer to the Development of Chemical Engineering and Industrial Chemistry, Paris, 1978.

[Abstract]

The equation slover EQUATRAN——An application of bipartite graph.
Goro Oguchi, Systems and Computing Dept. Mitsui Toatsu Chemicals Inc.
Bulletin of the Japan Society for Industrial and Applied Mathematics, 3 (1991), 213-229.

The procedure of manipulating simultaneous equations in EQUATRAN is presented. EQUATRAN is a computer program for getting the numerical solution of mathematical models including nonlinear equations, ordinary differential equations and other modeling elements. In this procedure, a bipertite graph, where one set of vertices represents valiables and another

set represents mathematical operations, is use to represents the equation system. It is shown that the bipertite graph is useful for checking the structual soluvability of the equation system, making the necessary transformation of equations and generating the optimum steps of the numerical calculation. A simple method generating Newton-Raphson process for solving simultaneous nonlinear equations based on automatic analitical differentiation is also proposed.

(1991 年 3 月 4 日受付)