非線形連立方程式の自動求根プログラム†

野口 悌三†† 安藤 裕昭††

非線形連立方程式の根を求めるためには，通常問題ごとにプログラムを作成しなければならないが，本論文は，任意に与えられた非線形連立方程式に対し，その根を求める計算手順を自動的に作成・実行するプログラムについて報告する．計算手順の作成は，すべての式を1演算子式に分解した後，その順序を入れ替えることによって行われる．ニュートン法による繰返し計算のループが必要な個所に自動的に挿入される．実際にこの計算手順を実行して解を求めるためには，ニュートン法の初期値を与えることが必要であるが，これは実際の問題に則して与えるほうが有利である．このため，本プログラムは，会話型プログラムになっており，みずから作成した計算手順をインタプリティブに実行し，初期値は実行時に要求する．本プログラムは，設計部門等における雑多な式の集合を，プログラミングをせずに解くための道具として実用に供されている．

1. まえがき

線形連立方程式を解くプログラムは，あらかじめ用意しておくことができるが，非線形連立方程式を解くプログラムは，通常問題が発生するごとに個々に作成せざるをえない．非線形方程式においては，問題が与えられるまで計算式自体が明らかでないのであるからやむをえないことであるが，これは，製造業における開発設計などのように，絶えず新しい非線形連立方程式に遭遇するような場合には，かなりわずらわしい問題である．

そこで筆者らは，任意の非線形連立方程式を，問題を与えるだけで自動的に解くことのできるプログラムを作成した．本論文は，このプログラムの機能と処理方法について述べるものである．

非線形連立方程式を解くためには，通常ニュートンラフソンの逐次近似法[1]が用いられる．本プログラムでもこれを用いるが，繰返し計算のループのかかり方は問題ごとに異なるので，与えられた計算式からループのかかり方を発見し適切な計算手順を組み立てる一種の自動プログラミングの機能が必要である．

このような自動プログラミングは，すでに1966年にHomerが試みている[2]．しかし，これは，たとえば $A=B+C$ という式があるときは，$B=A-C$，$C=A-B$ という式を同時に与えておき，そのなかから順次計算可能なものを選択して使うというもので，ループを構成する能力もなかった．

式の順序の入れ替えは，汎用シミュレーションプログラムCSMP[3]などでも行われている．また化学プロセスの定常値を求めるプログラムJUSE-GIFSにおいては，ループの発生を自動的に行い，グラフ理論を応用して，非線形連立方程式の簡素化を行っている[4]．

これらの方法では，式の順序の入れ替えを基本としており，与えられた式はそのまま計算するものとしている．

筆者らは，先に一般的な式の計算順序の入れ替えについて研究した際，すべての式を1演算子式に分解することによって，任意の算術式の集合を解くプログラムを試作した[5]．

その後，実際に作成したプログラムを設計部門で使用し，その経験によりいくつかの改良を行った．

本論文は，現在筆者らが実用に供している非線形方程式の自動求根プログラムについて述べるものである．2章にプログラムの機能，3章にその処理方法，4章に実用的観点から行った改良のポイントを述べる．

なお，本プログラムの開発思想については，文献6)に述べている．

2. プログラムの機能

2.1 問題領域と問題

n 個の変数の間に m 個の互いに独立な関数関係が存在するとする（$m \leq n$）．一般には，このとき，$n-m$ 個の変数に任意に値を指定することができる．残りの m 個の変数は，関数関係の組を解いて定められる．

† A Program for Automatic Solving of Systems of Nonlinear Equations by KENZO NOGUCHI and HIROAKI ANDO (Technical Systems Center, Technical Head Quarters, Mitsubishi Heavy Industries, Ltd.).

†† 三菱重工業(株)技術本部技術計算センタ

与えられた関数関係の組を**問題領域**，値を指定された変数を**既知変数**，残りの変数を**未知変数**と呼ぶことにする．

一般にある物理系の解析（強度解析，性能解析等）を行うときは，物理系の状態を表すと考えられる何らかの数学的モデルが作られる．問題領域は，この数学モデルに対応している．

一つの問題領域に対して，種々の既知変数のとり方がありうる．そして，既知変数の組を指定することにより，残りの未知変数を解くという**問題**が具体的に発生する．

本プログラムは，問題領域が四則演算と初等関数のみによって構成されているときに，任意に指定された変数について，数値的にこれを解くものである．

2.2 問題領域の与え方

問題領域を構成する関数関係は，FORTRAN と同様の形式で書かれた方程式の集合として与える．ただし式は FORTRAN と異なり，右辺から左辺へ代入するという意味はもっていない．たんに変数の間の関係を表すだけである．したがって，右辺と左辺を逆に書いても，両辺に演算子が含まれていてもさしつかえない．たとえば，次の三つの式は，まったく同一の意味をもつ．

（1） $A=B+C+D$,

（2） $B+C+D=A$,

（3） $A-B=C+D$.

演算子としては，加減乗除とべき乗，および FORTRAN がもっている初等関数のすべてを使用することができる．

かっこを用いて任意に複雑な式を書くことも許されている．式を一つにまとめて書いても，中間変数を用いていくつかに分けて書いても同じである．たとえば次の二つの書き方は，いずれも同じ問題を定義する．

（1） $A=B+C+D$

（2） $A=E+D$; $E=B+C$

式と式の間は，セミコロンで区切って与える．式の数や変数の数は，プログラムが自分で数えるので，とくに指示する必要はない．

すべての式を与え終わると，ドル記号を一つ与える．これによって，問題領域を与え終わったことがプログラムに知らされる．

2.3 変数リスト

式を与え終わった時点では，式に含まれている変数には，既知・未知の区別がない．これを指示するために，問題領域を与え終わったあとに変数リストを与える．

変数リストには，初めに既知変数の名前を並べる．この順序は，後にデータを与えるときの順序になるので，利用者が理解しやすい順序に並べるとよい．

既知変数以外の変数は，すべて未知であるとみなされるが，そのなかには出力する必要のないものもありうるので，出力したいもののみ既知変数のあとに並べて指示することとした．各変数名の間はコンマで，既知変数と出力変数のリストの間はセミコロンで区切る．変数リストの終りは，ドル記号を与えて示す．

たとえば，

$Y=A*X**2+B*X+C$ $

という問題領域が与えられているとき，変数リストを

A, B, C, X ; Y $

とすれば，二次式を計算して Y を出力せよ，という意味であり，

A, B, C, Y ; X $

とすれば，二次方程式を解いて X を出力せよ，という意味になる．

変数リストと式の関係について，形式的な制限は何もない．しかし，変数リストが与えられるとプログラムは内部に計算手順を作成するので，このとき計算手順が作れないような問題になっていれば（たとえば，既知変数の数が多すぎる場合など）エラーとなる．

2.4 データの与え方

プログラムは，既知変数と未知変数の区別が明らかになった時点で，問題領域を構成する方程式群を変形し，未知変数の値を計算するための計算手順を作り出す．これはプログラム内部に保存され，外には出力されない．

それが終わると，プログラムは既知変数にデータを与えるよう要求する．データは変数リストのなかにおける既知変数の名前と同じ順序に並べて与える．するとプログラムは，計算手順を実行して，出力変数の値を表示する．

以上が本プログラムの基本的機能である．

3. 処　　理

本プログラムの概略のフローチャートを，**図1**に示した．

3.1 式の分解

問題領域の構成式は，任意の複雑さをもっていてさしつかえないが，プログラムは最初にこれを1演算子

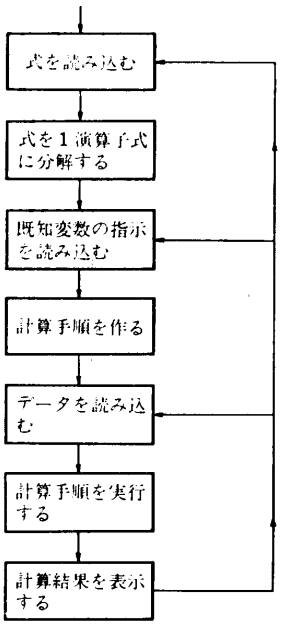


図 1　プログラムの概略フローチャート
Fig. 1　Flowchart of the program.

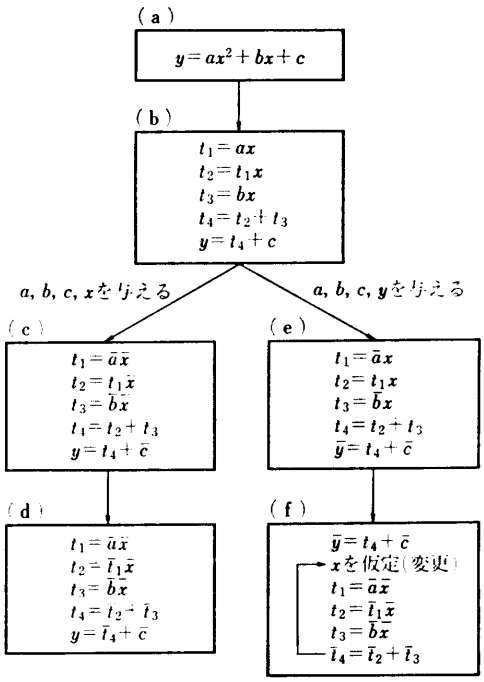


図 2　計算手順決定の例
Fig. 2　Example of generated calculation procedure.

の式に分解して蓄える．**図2**は，与えられた式が内部でどのように変形されていくかを示したものである．図2(a)は，与えられた式を示し，(b)は，これが1演算子式に分解された姿を示している．ここで t_i は挿入された中間変数である．

3.2　未知変数と既知変数の区別

図2(a)の式は，a,b,c,x を既知とすれば2次式の値を計算する問題であり，a,b,c,y を与えれば2次方程式を解く問題となる．図2(c),(e)は，このように，同じ式が与えられても，変数リストの与え方で計算手順が変わってくることを示している．

プログラムは，既知変数が指示されると，その変数に印をつける．図2では，変数名の上に横棒を引いてこれを示した．図2(c)は，a,b,c,x を既知とした場合，図2(e)は，a,b,c,y を既知とした場合を示している．

3.3　計算手順の決定

各式は，1演算子式であるので，左辺も含めて，2個または3個の変数を含んでいる．ここでは，定数は最初から既知の変数と考える．3個の変数を含む式とは，$a=b+c$ のような通常の二項演算子を含む式であり，2個の変数を含む式とは，$y=\sin(x)$ のような関数式である．

図2(c)または(e)のように，既知変数に印をつけたあと，プログラムは計算順序の入れ替えを行う．その部分の詳細フローチャートを**図3**に示したが，原理は次のとおりである．

まず未知変数を含む個数の最も少ない式を一つとりあげ，それが最初の式になるように式の順序を入れ替える．もし，その式が未知変数を1個しか含まないなら，その未知変数はその式で決定されると考え，以下の式ではその変数は既知であると考える．このプロセスを繰り返すと，未知変数を2個以上含む式ばかりになることがある．そのときは，適当に一つの変数に仮定値を与えるものとし，残りの式においてはこの変数を既知と考える．このようにして，先に進むと，今度はすべての変数が既知になってしまった式が残る．プログラムは，ここにループ命令を挿入する．これは，この式が自動的に満たされているか否かをチェックし，満たされていない場合は，仮定値を変更して再計算するよう指示するものである．

こうして生成された計算手順が，図2(d),(f)である．(f)には，ループが作られている．

3.4　計算手順の実行

本プログラムは，自分で作り出した計算手順をインタプリティブに実行する．

計算手順は，3種類の命令によって構成される．

一つは計算式である．これは，つねに一つの未知数

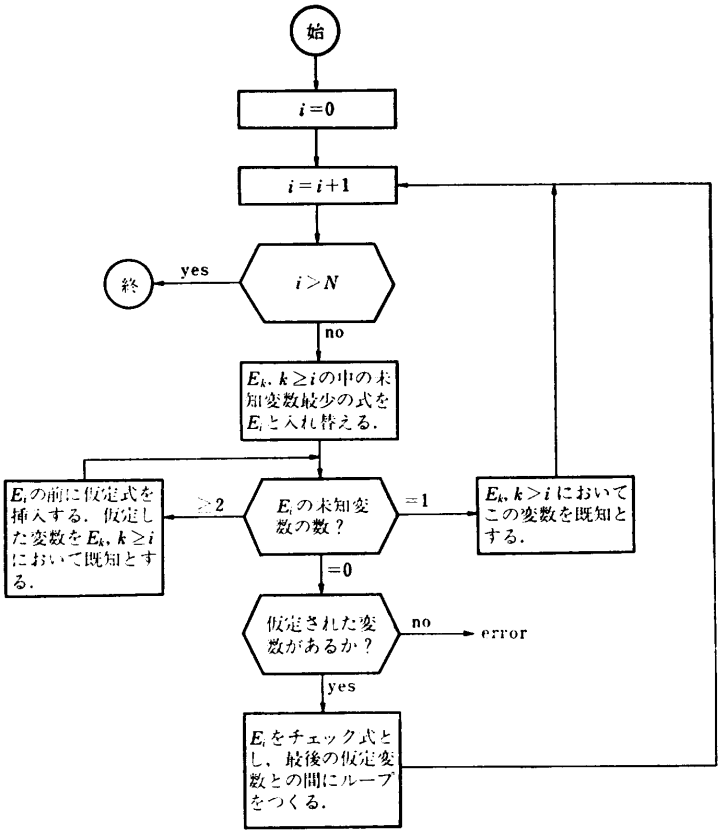


図 3 計算手順作成のフローチャート
Fig. 3 Flowchart of generating calculation procedure.

を含んでいるので，その値を決めるように計算が実行される．

たとえば

$\bar{a}=\bar{b}+c$

という式があれば，実際には

$c=\bar{a}-\bar{b}$

という演算が実行される．

注意すべきは，式の中に同一変数が含まれている場合である．

$\bar{b}=a+a$

$\bar{d}=c*c$

という式は，それぞれ

$a=\bar{b}/2$

$c=\sqrt{\bar{d}}$

と解かれる．

次は仮定命令である．これにぶつかると，プログラムは，初期値を要求し，与えられた値をその変数に与える．

最後はループ命令である．これには，つねに一つの式が対応している．この式をそのループの検定式と呼ぶ．検定式が，それまでの計算結果によって自動的に満たされていれば，プログラムはループを脱出する．

満たされていないときは，そのループに対応する仮定変数の値を修正して再び計算を繰り返す．これを行うのは，あらかじめ組み込まれたニュートン法のサブルーチンである．

ループと仮定変数の対応は，FILO(first-in last-out)型のスタックを用いることによって，自動的につけられる．

ニュートン法を使用するためには，微係数が必要であるが，これは数値微分によっている．すなわち，ループを制御するルーチンは，仮定変数の値をわずか変更して検定式の変化を求め，これから微係数の値を求め，ニュートン法を用いて仮定変数の値を推測し，再度仮定変数の値から検定式までの計算を繰り返す．これが1回のループに相当する．適当に定められた回数まで繰り返しても検定式が満たされないときは，プログラムは警告を発して計算を打ち切る．利用者は，このあと初期値を変えるなどして，新たに計算をやり直すことができる．

4. 実用化への配慮事項

4.1 関数の方向性

3.4 節に述べたように，未知変数を1個しか含まない式は，自動的にその変数について解いた形の計算を実行する．このことは，関数の場合にも同様である．たとえば，$\bar{y}=\sin(x)$ という式は，実際には $x=\sin^{-1}(\bar{y})$ と計算される．ただし，逆関数が多価になるものについては，あらかじめ主値が定めてある．

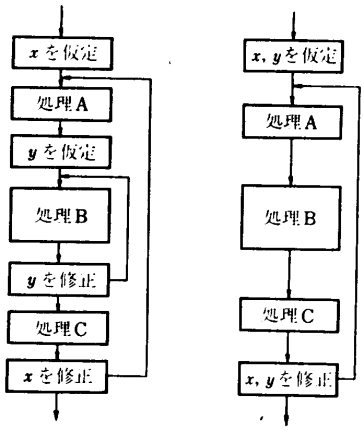
本プログラムは，FORTRAN がもっている関数はすべて利用可能としたが，そのなかには上記のような逆演算の不可能なものが存在する．たとえば，MAX, MIN, INT などである．

そこですべての関数を，両方向性のものと一方向性のものに区別して扱うこととした．一方向性の関数について逆方向に計算する必要が生じたときは，ループが余分にかかることとなる．

4.2 多変数ニュートン法

3.3 節に述べた計算手順作成の基本的手続では，ループが多重になったとき，内側のループから順次収束させていくこととなるため，多くの計算時間を必要とする．

そこで，多重ループが作られたときには，これを改造して，多変数ニュートン法を採用することとした．このためには，**図4** に示すように，ループの多重構造が確定した時点で，内側のループの仮定式を最外側ループの仮定式の位置まで繰り上げ，ループ命令挿入場所を最外側ループのループ命令の位置まで繰り下げればよい．もちろんこのときは，3.4 節に述べた数値微分の代りに，ヤコビアンを数値微分で求める必要がある．



(a) 多重ループ　　(b) 多変数ニュートン法のループ

図 4 多変数ニュートン法のループ生成
Fig. 4 Loop structure for multivariable Newton's method.

多変数ニュートン法を採用する前は，計算時間の点で本プログラムは実用にならなかったが，本改造により十分実用に供しうるものとなった．

4.3 収束性の問題

非線形方程式は，一般に複数の根をもっており，ニュートン法による逐次近似も，初期値によって望む値に収束したりしなかったりする．

これに自動的に対処するのは困難であるが，実際の問題では利用者に解の性質についての何らかの予想や知見があるのがふつうであり，利用者に初期値を与えさせることによって，上記の難点をほとんど回避できる．ただし，このためには，初期値を与える変数が利用者にとって意味のあるものでなければならない．そこで，方程式を1演算子式に分解するために挿入された中間変数は，初期値を与える変数には選ばないこととした．

4.4 走査計算機能と図化機能

走査計算機能というのは，変数にデータを与えるとき，一つの変数に複数の値を与えたり，出発値と増分を与えることによって，変数の値を系統的に変えて解の傾向を見ることのできる機能である．複数の変数にこの機能を用いて，マトリクス的に走査することもでき，また複数の変数を組にして動かしていくこともできる．

これに計算結果の図化機能を組み合わせて用いることにより，利用者は問題の全体の性質を把握しつつ計算を進めることができる．

5. 応用と評価

本プログラムは，主として機械設計における雑多な計算式の集合を，プログラミングを行わずに解くための道具として使われている．未知変数の数が数十個程度までなら十分実用可能である．

非線形方程式を通常のプログラムにすると，問題がわずかに変わっただけで計算手順を大きく変えなければならないことがあるが，本プログラムを利用すれば，そのような心配なしに問題を変更することができる．解析しようとする現象に対して数学モデルがまだ固まっていないときなどに便利である．

本プログラムは，与えられた式を1演算子式に分解しているため，与えられた式をそのままにして順序の

入れ替えだけを行うものよりも，ループの発生は少なくてすむ利点がある．

しかし，数式処理的な式の整理の機能はもっていないので，人手で解けばループなしに解けるものでもループが発生することがある．

プログラムはすべて FORTRAN で書かれており，基本的な部分は約2,000ステートメントであるが，最近はファイル関係の機能，図化機能等を充実させ，総合的設計システムとなっており，全体では14,000ステートメントに達している．

6. あとがき

本論文では，任意の非線形連立方程式に対して，自分で計算手順を発見して解を求めるプログラムについて述べた．

本プログラムの特徴は，与えられた式を1演算子の式に分解してから計算順序の入れ替えを行うことにより計算手順を作成していることで，これにより，たんに与えられた式の順序を入れ替えるだけのものより，繰返し計算のループの数を少なくしている．

参考文献

1) 情報処理学会：新版情報処理ハンドブック，p. 371 (1980)．
2) Homer, E. D.: An Algorithm for Selecting and Sequencing Statements as a Basis for a Problem-oriented Programming System, Proc. ACM National Meeting, pp. 305-312 (1966).
3) 日本アイ・ビー・エム株式会社：CSMP Ⅲ. user's guide, p. 29.
4) 日本化学技術研修所計算センタ：化学プロセスシミュレータ JUSE-GIFS プログラム利用の手引，第三版 (1970)．
5) 野口，安藤，三浦：汎方向性プログラム，三菱重工技報，Vol. 12, No. 3, pp. 1-5 (1975)．
6) 野口懐三：CAD システムのための問題向き言語，日本機械学会誌，Vol. 81, No. 713, pp. 66-71 (1978)．

(昭和57年5月12日受付)
(昭和57年9月 6 日採録)