JAERI-M
7552

連立非線形方程式の数値解法プログラム
（ＳＳＬの拡充とベンチマーク・テスト №6）

１９７８年３月

朝 岡 卓 見

日 本 原 子 力 研 究 所
Japan Atomic Energy Research Institute

JAERI－M 7552

連立非線形方程式の数値解法プログラム
（SSLの拡充とベンチマーク・テストNo.6）

日本原子力研究所東海研究所原子炉工学部
朝岡　卓見

（1978年1月30日受理）

実変数の非線形実関数の連立方程式の1つの解を，推定値から求める数値解法として，準Newton法と射影法とを概観し，それによる計算プログラムの整備とベンチマーク・テストを実施した。

ヤコビアン行列の微分形を与えなくてよいルーチンとしては，PowellのNS01A，ヤコビアンが疎な系を対象としたReidのNS03A，及びBrownのNONLINの3つが取り扱われた。これらはいずれも準Newton法を用いているが，この中ではNONLINが最も安定したアルゴリズムをもっており，計算時間も短くて優れている。

ヤコビアンの微分形を与えるルーチンとしては，Boggsのアルゴリズムによる準Newton法のINTECHと，Georg & Kellerによる射影法のPROJA，及びNS03Aの1つのオプション，の3つを取り扱った。この結果，射影法はまだ，最適なアルゴリズムを与えるには研究が必要だが，線形変数のみを別個に扱えるINTECHが，この中では一応計算時間の点でも優れていることが示された。

JAERI-M 7552

Computer Programs for Solving Systems of Nonlinear Equations

(Development and Benchmark Test of SSL, No. 6)

Takumi ASAOKA

Division of Reactor Engineering, Tokai Research Establishment, JAERI

(Received January 30, 1978)

Computer programs to find a solution, usually the one closest to some guess, of a system of simultaneous nonlinear equations are provided for real functions of the real arguments. These are based on quasi-Newton methods or projection methods, which are briefly reviewed in the present report. Benchmark tests were performed on these subroutines to grasp their characteristics.

As the program not requiring analytical forms of the derivatives of the Jacobian matrix, we have dealt with NS01A of Powell, NS03A of Reid for a system with the sparse Jacobian and NONLIN of Brown. Of these three subroutines of quasi-Newton methods, NONLIN is shown to be the most useful because of its stable algorithm and short computation time.

On the other hand, as the subroutine for which the derivatives of the Jacobian are to be supplied analytically, we have tested INTECH of a quasi-Newton method based on the Boggs' algorithm, PROJA of Georg & Keller based on the projection method and an option of NS03A. The results have shown that INTECH, treating variables which appear only linearly in the functions separately, takes the shortest computation time, on the whole, while the projection method requires further research to find an optimal algorithm.

Keywords: Nonlinear Simultaneous Equations, Scientific Subroutine Library, Computer Program, Benchmark Test, Quasi-Newton Method, Projection Method, Powell's Algorithm, Reid's Algorithm, Brown's Algorithm, Boggs' Algorithm, Algorithm of Georg & Keller.

総　　論

ここ数年来，数値解析理論や新しいアルゴリズムばかりでなく，SSL (Scientific Subroutine Library，科学用サブルーチン・ライブラリー）や，特定の問題により深く関係した数値解析プログラム（例えば，Comp. Phys. Comm.）の発表件数は，膨大なものがある。従ってこれらのルーチンを隈なくサーベイし，且つ整備しておくことは不可能であるのは勿論，効率的利用という立場から見ても，労多くして，功少なしの感がある。そこで実用上は，各ルーチンの特徴の分析やベンチマークテスト，"stiff"な問題への使用経験など集約された情報が要求され，これらが充分蓄積された時に，ルーチン相互の位置づけや体系化が可能となる。

我々の研究室では，数値解析の広い分野にわたる，これまでのアルゴリズム調査を土台としてベンチマークテストなどの情報集約と評価を行った。対象としたのは，主に最近発表されたプログラムとアルゴリズムで，先に行ったアルゴリズム調査の延長という形でまとめたので御利用いただきたい。

（西田雄彦，藤村統一郎）

目　　次

1. 序　論 …… 1
2. 準Newton法ルーチン …… 1
3. 射影法（projection method）ルーチン …… 18
4. ベンチマーク・テストと検討 …… 21
5. 結　論 …… 33
謝　辞 …… 35
参考文献 …… 36
付録1　NS01Aルーチンとテスト用プログラムの FORTRAN リスト …… 38
付録2　NS03Aルーチンとテスト用プログラム（HMAX=1）の FORTRAN リスト …… 42
付録3　NS03Aルーチンのテスト用プログラム（HMAX=0）の FORTRAN リスト …… 53
付録4　NONLINルーチンとテスト用プログラムの FORTRAN リスト …… 54
付録5　INTECHルーチンとテスト用プログラムの FORTRAN リスト …… 57
付録6　PROJAルーチンのテスト用プログラムの FORTRAN リスト …… 61

CONTENTS

1. Introduction 1

2. Subroutines of Quasi-Newton Methods 1

3. Subroutines of Projection Methods 18

4. Benchmark Tests and Discussions 21

5. Conclusions 33

Acknowledgments 35

References 36

Appendix 1 FORTRAN Lists of NS01A Subroutine and its Test Program 38

Appendix 2 FORTRAN Lists of NS03A Subroutine and its Test Program for HMAX=1 42

Appendix 3 FORTRAN List of Test Program for HMAX=0 of NS03A Subroutine 53

Appendix 4 FORTRAN Lists of NONLIN Subroutine and its Test Program 54

Appendix 5 FORTRAN Lists of INTECH Subroutine and its Test Program 57

Appendix 6 FORTRAN List of Test Program for PROJA Subroutine . 61

1. 序　　論

代数方程式，超越方程式の数値解法のうち，高次代数方程式，すなわち多項式のすべての根を求めるための，逆補間法，Newton-Raphson法，Bairstow法，Jarrat法によるサブルーチンについては，すでに報告がまとめられている。[1] また，線形代数方程式，すなわち連立一次方程式を解くプログラムについても，別途まとめられているので，[2] 本報では連立非線形方程式の数値解法プログラムについて述べる。

連立非線形方程式

$$f_j(x_1, x_2, \cdots, x_n) = 0, \quad j = 1 \sim n \tag{1}$$

の解を求める問題にはしばしば遭遇する。本報では実変数xの実関数fのみを扱うことにする。(1)式はベクトル的に書くと

$$\underline{f}(\underline{x}) = 0 \tag{2}$$

となるが，1変数の1つの関数に対しても，前報でみたように，[1] 必ずしもすべての解を見付ける手法が確立されているとはいえない。連立方程式の際には，1つの推定値から出発して，1つの解を求めるのが精一杯の現状である。一般に解法としては，近似解の近くで(2)式を線形方程式で近似することにより，次の近似解を求める反復法が有用である。前報で述べたNewton-Raphson法は，まさにその典型的な手法であるが，これを実用に適するように改良した準Newton法による数値解法ルーチンをまず次章で述べる。次いで第3章で連立一次方程式の一解法である射影法を非線形系へ適用したルーチンについて述べ，第4章でこれらサブルーチンのベンチマーク・テスト結果の検討を行う。付録には，これらプログラムのFORTRANリストを，テスト用の簡単な例題と共に示しておいた。

2. 準Newton法ルーチン

Newton法は一般に(2)式に対して以下のように書ける。

$$\underline{x}_{i+1} = \underline{x}_i - \underline{A}_i^{-1}\underline{f}(\underline{x}_i) \tag{3}$$

ここで$\underline{A}_i$はi番目の反復近似解$\underline{x}_i$でのヤコビアン行列$\partial f_j / \partial x_k$である。これの実用上の問題は$\underline{A}_i$の計算である。たとえこの偏微分が解析的に求められてもn^2回の計算は大変で，数値的に求める際にはベクトル関数$\underline{f}$を，独立変数の少くとも$(n+1)$セットに対して計算しなければならない。もう1つの問題は，反復のための良い初期値を見付けることがしばしばむづかしく，このため収束しなくなることである。

この改良として，ヤコビアン行列は各反復ステップでなく，2～3回毎に計算するとか，1度

1. 序　　論

代数方程式，超越方程式の数値解法のうち，高次代数方程式，すなわち多項式のすべての根を求めるための，逆補間法，Newton-Raphson法，Bairstow法，Jarrat法によるサブルーチンについては，すでに報告がまとめられている。[1] また，線形代数方程式，すなわち連立一次方程式を解くプログラムについても，別途まとめられているので，[2] 本報では連立非線形方程式の数値解法プログラムについて述べる。

連立非線形方程式

$$f_j(x_1, x_2, \cdots, x_n) = 0, \quad j = 1 \sim n \tag{1}$$

の解を求める問題にはしばしば遭遇する。本報では実変数xの実関数fのみを扱うことにする。(1)式はベクトル的に書くと

$$\underline{f}(\underline{x}) = 0 \tag{2}$$

となるが，1変数の1つの関数に対しても，前報でみたように，[1] 必ずしもすべての解を見付ける手法が確立されているとはいえない。連立方程式の際には，1つの推定値から出発して，1つの解を求めるのが精一杯の現状である。一般に解法としては，近似解の近くで(2)式を線形方程式で近似することにより，次の近似解を求める反復法が有用である。前報で述べたNewton-Raphson法は，まさにその典型的な手法であるが，これを実用に適するように改良した準Newton法による数値解法ルーチンをまず次章で述べる。次いで第3章で連立一次方程式の一解法である射影法を非線形系へ適用したルーチンについて述べ，第4章でこれらサブルーチンのベンチマーク・テスト結果の検討を行う。付録には，これらプログラムのFORTRANリストを，テスト用の簡単な例題と共に示しておいた。

2. 準Newton法ルーチン

Newton法は一般に(2)式に対して以下のように書ける。

$$\underline{x}_{i+1} = \underline{x}_i - \underline{A}_i^{-1}\underline{f}(\underline{x}_i) \tag{3}$$

ここで$\underline{A}_i$はi番目の反復近似解$\underline{x}_i$でのヤコビアン行列$\partial f_j / \partial x_k$である。これの実用上の問題は$\underline{A}_i$の計算である。たとえこの偏微分が解析的に求められてもn^2回の計算は大変で，数値的に求める際にはベクトル関数$\underline{f}$を，独立変数の少くとも$(n+1)$セットに対して計算しなければならない。もう1つの問題は，反復のための良い初期値を見付けることがしばしばむづかしく，このため収束しなくなることである。

この改良として，ヤコビアン行列は各反復ステップでなく，2～3回毎に計算するとか，1度

だけしか計算しないなどの方法が考えられる。Barnesは，ヤコビアンの近似値とこれに対する補正を，各繰り返しステップで関数値を計算した後に行うアルゴリズムを発表している。[3] すなわち，

$$\underline{x}_{i+1} = \underline{x}_i + \delta\underline{x}_i, \quad \delta\underline{x}_i = -\underline{B}_i^{-1}\underline{f}(\underline{x}_i),$$

$$\underline{B}_{i+1} = \underline{B}_i + \underline{f}(\underline{x}_{i+1})\,\underline{u}_i^{T}/[\underline{u}_i^{T}\,\delta\underline{x}_i] \tag{4}$$

とし，これによりヤコビアンが以下の式を満足するようにしている。

$$\underline{f}(\underline{x}_{i+1}) = \underline{f}(\underline{x}_i) + \underline{B}_{i+1}\,\delta\underline{x}_i \tag{5}$$

$\underline{u}_i$は，線形系に対しては$\underline{x}_{n+2}$が解となるように，$i \geq n$ならばその前のn+1ステップの$\delta\underline{x}_{i-n+1} \sim \delta\underline{x}_{i-1}$と直交するように，$i < n$ならば$\delta\underline{x}_1 \sim \delta\underline{x}_{i-1}$に直交するように選んでいる。すなわち(4)式より

$$\underline{B}_{i+1}\,\delta\underline{x}_j = \underline{B}_{j+1}\,\delta\underline{x}_j, \quad 1 \leq i-j < n \tag{6}$$

あるいは，$i \geq n$に対しては

$$\underline{B}_i\,\delta\underline{x}_{i-j} = \underline{f}(\underline{x}_{i-j+1}) - \underline{f}(\underline{x}_{i-j}), \quad j = 1 \sim n \tag{7}$$

(7)式は$\underline{B}_i$が(n+1)番目の$\underline{x}$と$\underline{f}$で定義される線形系$\underline{f}(\underline{x}) = \underline{A}'\underline{x} - \underline{b}$の$\underline{A}'$の値になっていること，及びこの方法が一般化されたsecant法になっていることを示している。

収束については，Newton-Raphson法が2位の速さであるに対し，解を$\underline{\alpha}$とすると

$$|\underline{x}_{i+n+1} - \underline{\alpha}| = c\,|\underline{x}_i - \underline{\alpha}| \cdot |\underline{x}_{n+i} - \underline{\alpha}| \tag{8}$$

である。

同様な方法として，Broydenは，ヤコビアンの逆行列の近似値への補正を$\underline{f}$から求めていく方法を提案している。[4] すなわちNewton法のアルゴリズム(3)を

$$\underline{x}_{i+1} = \underline{x}_i + t_i\,\underline{p}_i, \quad \underline{p}_i = -\underline{B}_i^{-1}\underline{f}(\underline{x}_i) \tag{9}$$

とし，発散を防ぐように選ばれる因子t_iを導入している。$\underline{x}$をtのみの関数と考えれば$d\underline{f}/dt = \underline{A}\underline{p}_i$となるので，$\underline{f}$のTaylor展開の第1項を用いると，$\underline{A}$の近似は

$$\underline{f}(t_i) - \underline{f}(t_i - s_i) = s_i\,\underline{B}_{i+1}\,\underline{p}_i \tag{10}$$

従って(9)の第1式の反復は以下と組み合わされる。

$$\underline{H}_{i+1}\,\underline{u}_i = s_i\,\underline{p}_i, \quad \underline{p}_i = -\underline{H}_i\,\underline{f}(\underline{x}_i),$$

$$\underline{H}_i = \underline{B}_i^{-1}, \quad \underline{u}_i = \underline{f}(t_i) - \underline{f}(t_i - s_i) \tag{11}$$

(9)の第1式より，$\underline{f}(t_i) = \underline{f}(\underline{x}_{i+1})$，又$s_i = t_i$（殆んどすべての場合にこのように選ばれる）で$\underline{f}(t_i - s_i) = \underline{f}(\underline{x}_i)$なので，(11)式のように反復ステップ毎に$\underline{H}_i$は変るが，$\underline{f}(\underline{x})$の計算は特別には不必要である。$s_i = t_i$と選ぶと，$\underline{H}_{i+1}$は以下のように書ける。[5]

$$\underline{H}_{i+1} = \underline{H}_i - \underline{H}_i \underline{u}_i \underline{v}_i{}^T + \underline{p}_i t_i \underline{q}_i{}^T,$$

$$\underline{q}_i{}^T \underline{u}_i = \underline{v}_i{}^T \underline{u}_i = \underline{1} \tag{12}$$

結局アルゴリズムとしては,[6] ①解の近似$\underline{x}_0$を求める。②$\underline{x}_0$でのヤコビアンを数値的にでも求め，これの逆行列として$\underline{H}_0$を求める。③$\underline{f}(\underline{x}_i)$を計算。④$\underline{p}_i = -\underline{H}_i \underline{f}(\underline{x}_i)$を計算。⑤$\underline{x}_{i+1} = \underline{x}_i + t_i \underline{p}_i$のとき$\underline{f}$のユークリード・ノルムが$|\underline{f}(\underline{x}_{i+1})| < |\underline{f}(\underline{x}_i)|$をみたすように$t_i$を選ぶ。⑥$|\underline{f}(\underline{x}_{i+1})|$が収束しているかテストし，⑦収束していなければ$\underline{u}_i = \underline{f}(\underline{x}_{i+1}) - \underline{f}(\underline{x}_i)$を計算。⑧

$$\underline{H}_{i+1} = \underline{H}_i - (\underline{H}_i \underline{u}_i - \underline{p}_i t_i) \underline{p}_i{}^T \underline{H}_i / (\underline{p}_i{}^T \underline{H}_i \underline{u}_i) \tag{13}$$

を計算。⑨ステップ④以後を繰返す。しかしt_iを⑤のようにえらぶと収束を妨げる場合があり，むしろ良い$\underline{x}_0$と$\underline{H}_0$が与えられていれば，$t_i = 1$とおく方がよい。

Shampine & Gordon は,[7] $t_i = 1$とした(13)式によりヤコビアンの逆行列の近似値を補正していくプログラムQNを開発しているが，プログラムは公開されていないようである。なお，(9)式に基づく一解法のFORTRANプログラムは，最近McCueにより発表されていることを付言しておく。[8]

Broydenの(9)式の考えを更に進めたのがPowell[9]である。非線形代数方程式を扱っているが，Newton法はむしろ$\underline{x}_i$に対する補正方向$\underline{p}_i$を求めるのに用い，

$$||\underline{f}(\underline{x}_i + t\,\delta\,\underline{x}_i)|| < ||\underline{f}(\underline{x}_i)||, \tag{14}$$

$$||\underline{f}|| = \sum_{j=1}^{n} (f_j)^2.$$

となるようにパラメーターtを決めている。しかしこのHaselgrove法は,[10] ヤコビアンが特異（singular）のときは使えないので，(9)式を最小自乗形式にし，

$$\underline{B}_i{}^T \underline{B}_i\, \delta\, \underline{x}_i = -\underline{B}_i{}^T \underline{f}(\underline{x}_i) \tag{15}$$

これより単位マトリックスIを用いて

$$[\,t^* I + \underline{B}_i{}^T \underline{B}_i\,]\, \delta \underline{x}_i{}^* = -\underline{B}_i{}^T \underline{f}(\underline{x}_i) \tag{16}$$

により，パラメーターt^*を変えて

$$||\underline{f}(\underline{x}_i + \delta\, \underline{x}_i{}^*)|| < ||\underline{f}(\underline{x}_i)|| \tag{17}$$

としている。このLevenberg／Marquardt法は，$||\underline{f}||$の停留点以外は使える。それ故，この方法は$||\underline{f}||$の最急降下の方向にステップをとる考えになっているが，収束点が果して$||\underline{f}||$の大域的最小になっているかは分らない。なおヤコビアン行列の扱いはBroyden[4)5)]に基づいて効率良くしている。FORTRANプログラム・リストと，それの詳細な説明も付けられているので，我々はこのNS01Aルーチンを整備した。

このプログラム中の計算の流れをFig. 1に示したが，その第1ページの中央付近で$\underline{H}\underline{f}(\underline{x}) = \underline{B}^{-1}\underline{f}(\underline{x})$の絶対値が$\Delta$より小さければNewton法を用い，そうでなければ最急降下の方向に$\delta\,\underline{x}$

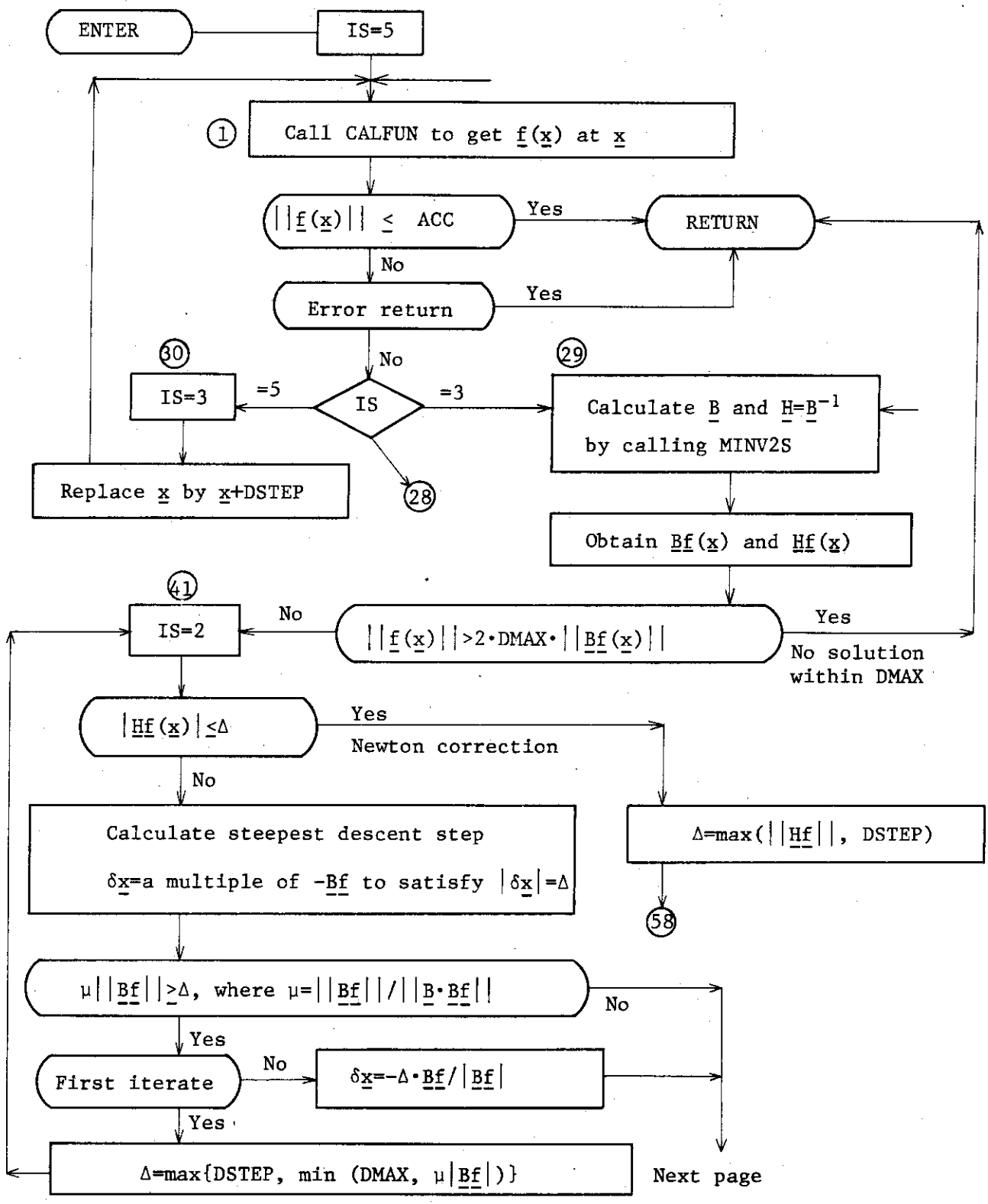

Fig. 1 Flow diagram of NS01A to solve $\underline{f}(\underline{x})=0$ from an estimate of $\underline{x}$

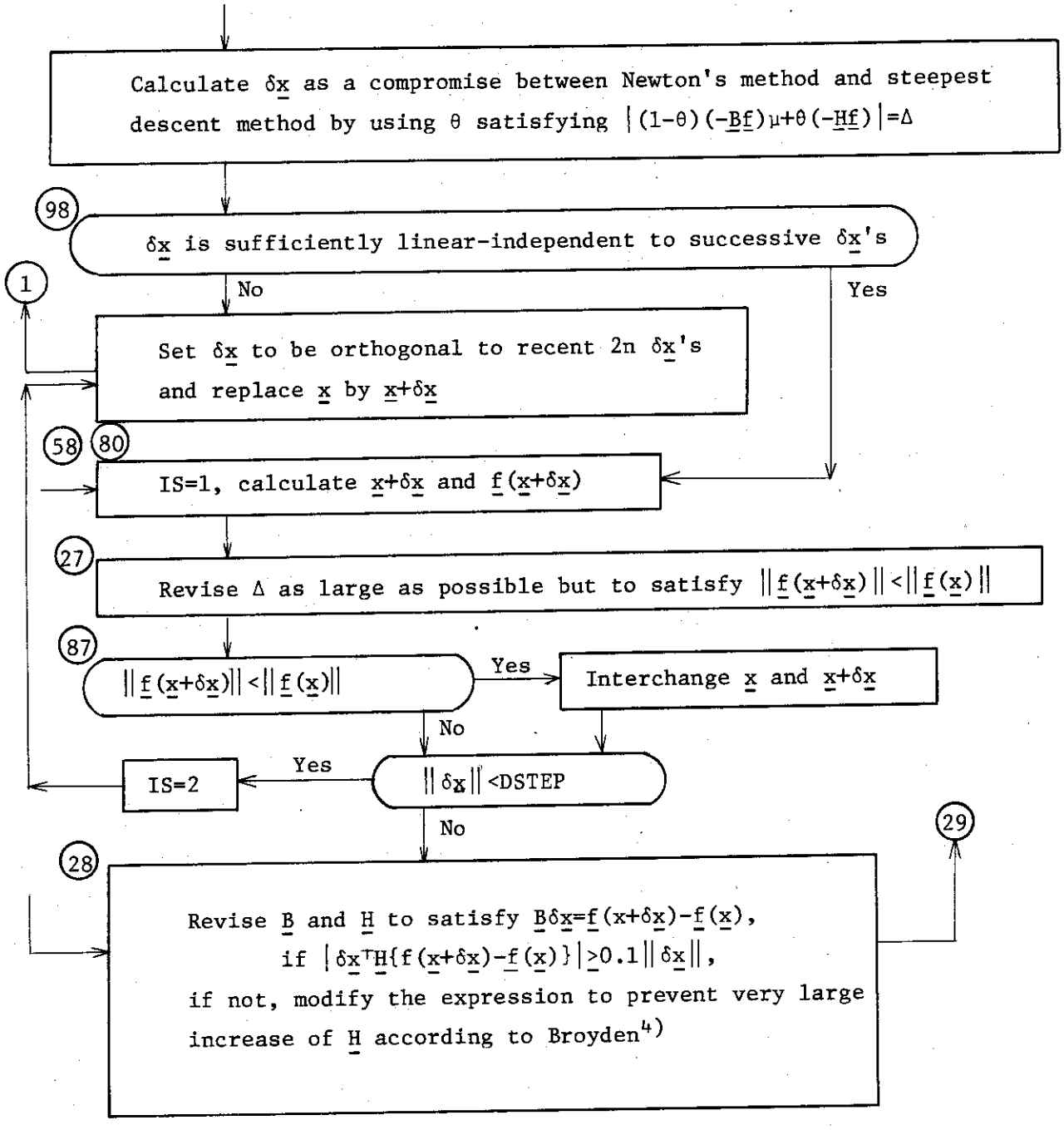

Fig. 1 (Continued)

をとっている。ここでΔは，常に(17)式が成立するように，Fig. 1 の第1ページ下部で選ばれている。なおヤコビアンは2ページ目の下方で，(5)式を満足するように$\underline{B}$，及びHを$\underline{B}$roydenのアルゴリズム[4]に従って補正している。このルーチンNS01AのFORTRANプログラム・リストを，テスト用のメイン・プログラム，及び関数値を求めるCALFUNルーチンと共に付録1に示した。

このNS01Aの呼び出しは以下によりなされる。

CALL NS01A(N, X, F, AJINV, DSTEP, DMAX, ACC, MAXFUN, IPRINT, W)

ここでNは方程式の数n（＞1）であり，Xは大きさNの1次元配列で，解の推定値が入力され，計算結果の解が出力される。DSTEP（付録1のメイン・プログラムではSTEP）は関数値から1階微分の近似値を求めるために使われる十分小さいステップで以下のように用いられる。

$$\partial f_1(\underline{x})/\partial x_1 = [f_1(x_1+\mathrm{DSTEP}, x_2, \cdots, x_n) - f_1(\underline{x})]/\mathrm{DSTEP} \qquad (18)$$

DSTEPはすべての変数に共通して使われる。DMAXは解の推定値と真値との間のユークリッド空間での推定距離で，各反復での$\delta\underline{x}$の上限として用いられると共に，$\underline{f}(\underline{x})$の勾配が小さくてDMAXより非常に遠くに解をもっていくようになった時，エラーとしてルーチンから戻るためにも使われている。この過程はFig. 1のフロー・チャートの第1ページの中央付近に示されている。ACCは解に要求する精度で，(14)式の$||\underline{f}||$をACC以下にするので，$f_j(\underline{x})$がすべてのjに対して同程度の大きさをもつようにscaleすることが望ましい。MAXFUNは$f_j(\underline{x})$，j=1～n，の計算回数による打ち切りに用いられる。一般に10n回程度の計算が要求されるので，これより大きい値を入力すればよい。IPRINTはプリント出力に関し，0の際にはエラーメッセージのみ，1の際には$\underline{f}(\underline{x})$の計算毎に$\underline{x}$と関数値を出力する。Fは大きさN以上の1次元配列で，一般に$\underline{f}(\underline{x})$のストアに使われ，AJINVはNxNの2次元配列で，ヤコビアンの逆行列が計算終了時にセットされ，必要に応じ(15)式に従い$\underline{x}$の誤差が$-\underline{B}^{-1}\underline{f}(\underline{x})$によって計算できるようになっている。Wは$n(2n+5)$の大きさの1次元配列の作業領域で，計算終了時にはヤコビアンの値が最初のn^2にセットされる。なお$\underline{f}(\underline{x})$の形は，付録1の例のようにCALFUN(N, X, F)で定義されなくてはならない。N, X, FはNS01Aと同様で，F(j)に$f_j(x)$が与えられる。

なお上述のDSTEPとMAXFUNによる計算打ち切りに加え，$||\underline{f}||$が連続した(n+4)回の反復でも減少しない時，及び新しくヤコビアンが計算された後でも減少しない時にもエラーとしてNS01Aから戻される。逆行列の計算のためには，FACOM SSL[11]のMINV2S(AJINV, KI, N, EPS, ILL)が，付録1のNS01AルーチンのISN=92で呼ばれている。入力行列がAJINV(KI, KI)に入れられ，演算後の逆行列がAJINV(N, N)にセットされる。EPSは掃き出し法で解く時の軸の値がこれ以下になると誤差が大きくなるので演算を打ち切るためで，単精度なので10^{-5}～10^{-30}が標準値である。ILLはMINV2Sから正常に戻った時は0，入力パラメータのエラーの時30000，入力行列の絶対値がEPS以下の時29000，それ以外の値の時は，軸がEPS以下になって演算を打ち切った時のスイープアウトの回数である。

このNS01Aルーチンと同様，Marquardtのアルゴリズムで，ヤコビアン行列が疎(sparse)のときを，Reid[12]は(16)式の右辺をCholesky分解し，

$$[t^*\mathrm{I} + \underline{B}_i^{\mathrm{T}}\underline{B}_i]\,\delta\underline{x}_i^* = \underline{L}_i\,\underline{L}_i^{\mathrm{T}} \qquad (19)$$

を前進消法及び後退代入法で解いている。t^*は(17)式の両辺の差と，

$$\underline{f}(\underline{x}_i+\delta\underline{x}_i^*)=\underline{f}(\underline{x}_i)+\underline{B}_i\,\delta\underline{x}_i^* \tag{20}$$

と線形近似した時の補正

$$t^*(\delta\underline{x}_i^*)^T\delta\underline{x}_i^*-\underline{f}^T(\underline{x}_i)\underline{B}_i\,\delta\underline{x}_i^* \tag{21}$$

を比較し，この両者の比が1に近いときには(20)式の線形近似が成立するので，t^*を減少して長いステップ$\delta\underline{x}_i^*$をとり，そうでなければ逆に短いステップをとっている。又ヤコビアンの近似$\underline{B}_i$は，求められた$\delta\underline{x}_i^*$に対して(20)式を満足し，かつ疎の性質が保存されるように，それぞれの行について以下の式により補正を行っている。

$$\delta\underline{B}_i=(\underline{u}_i-\underline{B}_i\,\delta\underline{x}_i^*)(\delta\underline{x}_i^*)^T/[(\delta\underline{x}_i^*)^T\delta\underline{x}_i^*],$$

$$\underline{u}_i=\underline{f}(\underline{x}_i+\delta\underline{x}_i^*)-\underline{f}(\underline{x}_i) \tag{22}$$

このアルゴリズムによるFORTRANプログラム・リストと，その詳細な説明も記されているので，これの倍精度ルーチンNS03Aの整備も行った。

このルーチンは付録2に示したテスト用のメイン・プログラムにみられるように以下により呼び出される。

CALL NS03A(QUNC, M, N, X, SAC, STPMIN, MAXFUN, IPRINT, W, IW, IRN, IP, A, IRNA, IPA, HMAX)

ここでQUNCは原典と異なりダミーとなっており，Mは方程式の数，Nは変数の数，Xは大きさNの1次元配列で，入力としては近似解，出力としては求められた解がセットされる。SACは$||\underline{f}||$に要求する精度で，$||\underline{f}||\leqq$SACとなるまで計算がなされる。SAC＝0の時には，計算の終了は，$\underline{x}$に対する要求精度STPMINのみによる。$|\delta\underline{x}|<$STPMINとなるまで計算が行われるが，$|\delta\underline{x}|\leqq$EPS＊$|\underline{x}|$(EPS＝$1\times10^{-14}$)でもコントロールされるので，STPMIN＝0とセットしてもよい。MAXFUNは

$$\underline{f}(\underline{x})=\underline{r}(\underline{x})+\underline{A}\underline{x} \tag{23}$$

と書いた時の$\underline{r}(\underline{x})$を定義するサブルーチンFUNC(N, X, F, M, D)と呼ぶ上限回数である。FUNC中のN, X, MはNS03Aと同じで，大きさMのFには$\underline{r}(\underline{x})$が付録2のようにストアされる。$\partial r_j/\partial x_k$が解析的に求められる時(HMAX(1)＝0)には，付録3に示したように0でない微分がDの配列にストアされる。

NS03Aの引数のIPRINTはプリント出力の制御用で，0の際はエラー・メッセージのみ，0でない場合は最初と最後の結果，及びその間の|IPRINT|毎の反復の結果がプリントされる。IPRINT＞0の際は$||\underline{f}||$, $|\underline{x}|$, $|\delta\underline{x}|$, $|\underline{B}^T\underline{f}|$が，IPRINT＜0の際には，これに加え，$\underline{x}$，$\underline{f}$，$\underline{B}^T\underline{f}$も出力される。引数Wは大きさIWの配列で，(3M＋N)個の変数，$\underline{A}\neq\underline{0}$の時はM個の変数，$\partial r_j/\partial r_k$の0でない値，この微分形が与えられていない場合(HMAX(1)≠0)には(N＋1)個の変数，及び後述のMA17ルーチンでの3角型の正規行列とそれに付随する整数行列が入

れられる。

IRNとIPは，$\partial r_j/\partial x_k$のsparsityの形を与える整数配列で，0でない微分を，まずx_1による微分，次にx_2によるものと並べたとき，IP(K)がx_kによる微分の最初の位置で，IRN(J)がJ番目のr_jのjを表わしている。IPの配列の大きさは(N+1)で，IP(N+1)−1が0でない微分の総数を与え，これがIRNの配列の大きさになっている。この整数配列は後述のTD02Bルーチンを呼ぶことにより求めることもできる。Aは(23)式の$\underline{A}$の0でない要素a_{jk}が，まずk＝1のものから順に入れられる。IRNAとIPAがa_{jk}のsparsityを与える整数配列で，$\partial r_j/\partial x_k$に対するIRN，IPと同様に定義されている。$\underline{A}=\underline{0}$の時にはIPA(1)＝0とすればよい。HMAX(1)はすでに述べたように，$\partial r_j/\partial x_k$がFUNCに定義されている時に0，そうでなければ差分による計算の際のステップの上限を入力する。

NS03Aでの計算が成功裡に終った後，NS03Aのentryを

CALL NS03F(I,V)

により呼ぶことにより，variance-covariance行列$(\underline{B}^T\underline{B})^{-1}$を計算することができる。Iは行列の列で，大きさNの配列Vに値がセットされる。

NS03Aルーチン中の計算の流れはFig. 2に示した。これはメイン・サブルーチンで，Fig. 3に書かれているように多くの補助サブルーチンを呼んでいる。NS03C，TD02Aルーチン中の計算の流れはそれぞれFig. 4と5に示した。一方MC09A(M，N，A，X，Y，TRANS，IRN，IP)は，疎行列$\underline{A}$，ベクトル$\underline{x}$，$\underline{y}$に対して，TRANS＝falseの時は$\underline{A}\underline{x}+\underline{y}$，trueの時は$\underline{A}^T\underline{x}+\underline{y}$を計算するルーチンである。ここでMが$\underline{A}$の行数，Nが列の数で，Aに$\underline{A}$の0でない要素が，NS03Aの引数Aと同様に入れられる。Xには$\underline{x}$，Yには$\underline{y}$が入り，計算結果はYにセットされる。IRN，IPはNS03AのIRNA，IPAと同様に$\underline{A}$のsparsityの形を与える配列である。一方MC02AS(A，B，S，N)は2つのベクトル$\underline{a}$と$\underline{b}$の内積計算ルーチンで，Aに$\underline{a}$，Bに$\underline{b}$が入れられ，Sに結果がセットされる。Nは各ベクトルの次元数である。

Fig. 4にフローチャートを示したNS03Cは，Fig. 3からも分るように対称正定値疎行列の線形方程式系を解くMA17Aルーチン[13]を呼んでいる。このルーチンは，すでにJSSL[14]に登録されているが，MA17Aが使っているKB10AS(ITAB，INDX，N)は，大きさNのITABの列を，ITABの大きさの順に並べるために新しく作成されたルーチンである。付録2のプログラム・リストから分るように，大きさNの整数の列INDXを，ITAB(INDX)が小さい方から順に並べられるよう，すなわち，ITAB中のI番目に小さいものがITAB(INDX(I))になるように順序付けがなされる。

Fig. 5に計算の流れを示したTD02Aルーチンは，(23)式の$\underline{r}(\underline{x})$のヤコビアン行列を，要素が0の値になる位置が分っている状態で有限差分をとって計算するルーチンである。この際の差分のステップは入力で与えられた範囲内で自動的に調整される。このTD02Aには，Fig. 5にも示されているように2つのentry，TD02B，TD02Cがあるが，これらはそれぞれ以下により呼び出される。

CALL TD02A (M, N, IRN, IP, QUNC, H, X, Y, F, HMAX, A, IG, W, Z)
CALL TD02B (M, N, IRN, IP, QUNC, H, X, Y, F, W, IA)
CALL TD02C (N, IRN, IP, IA, MBD)

ここで，M，N，IRN，IP，QUNCはNS03Aの引数と同じで，Mがヤコビアンの行数，Nが列の

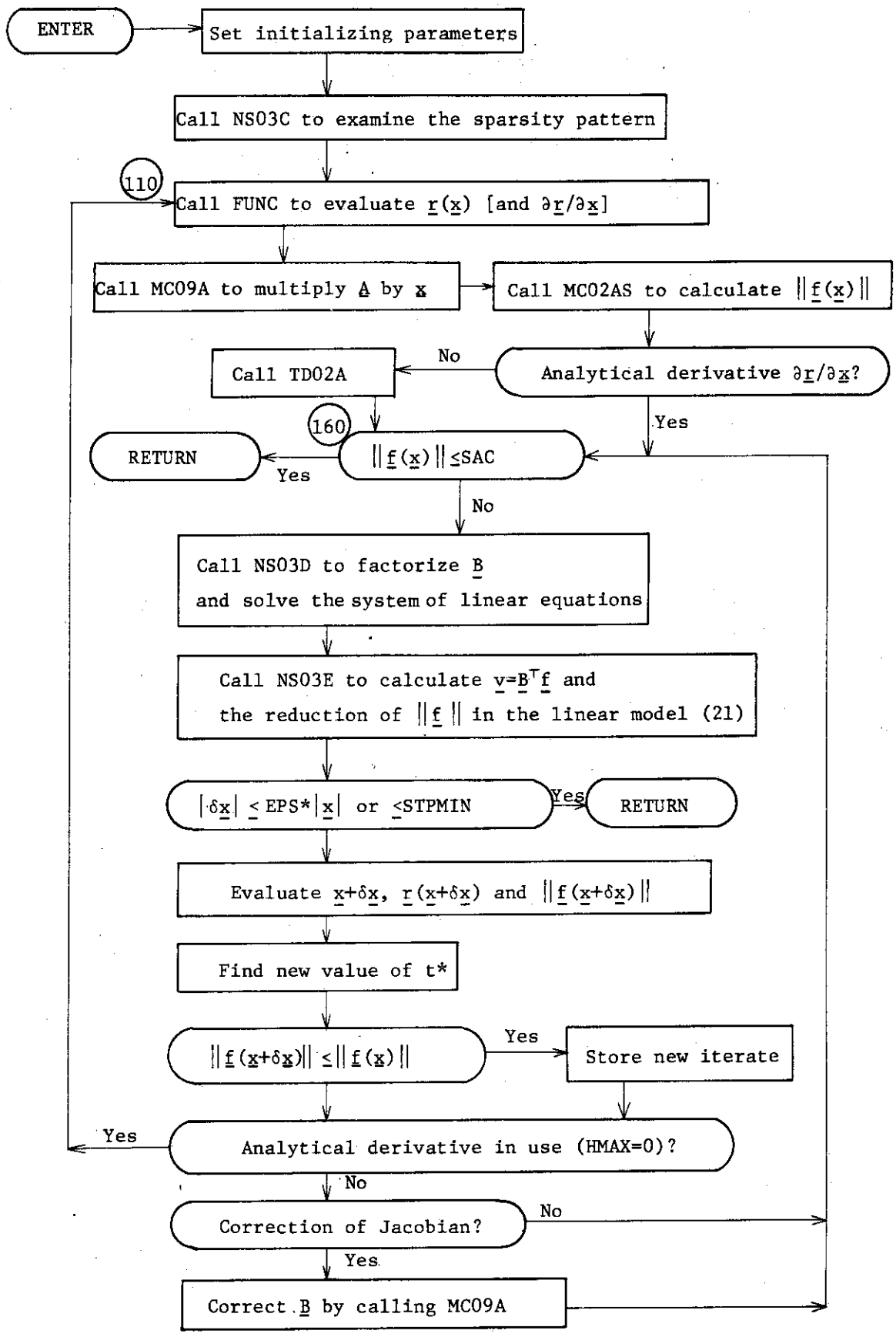

Fig. 2 Flow diagram of NS03A to solve $\underline{f}(\underline{x})=\underline{r}(\underline{x})+\underline{A}\underline{x}$ from an estimate of $\underline{x}$ (with the entry NS03F at the end)

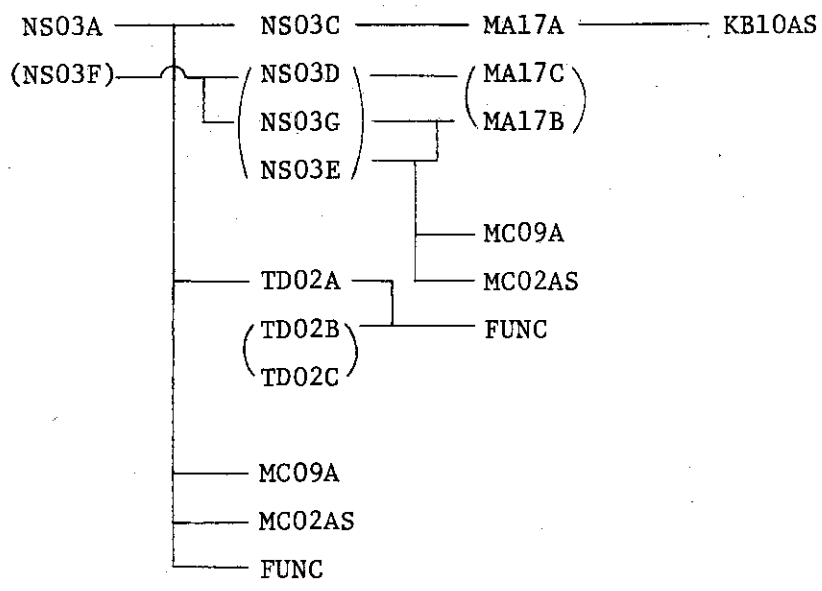

Fig. 3 Hierarchic structure of NS03A

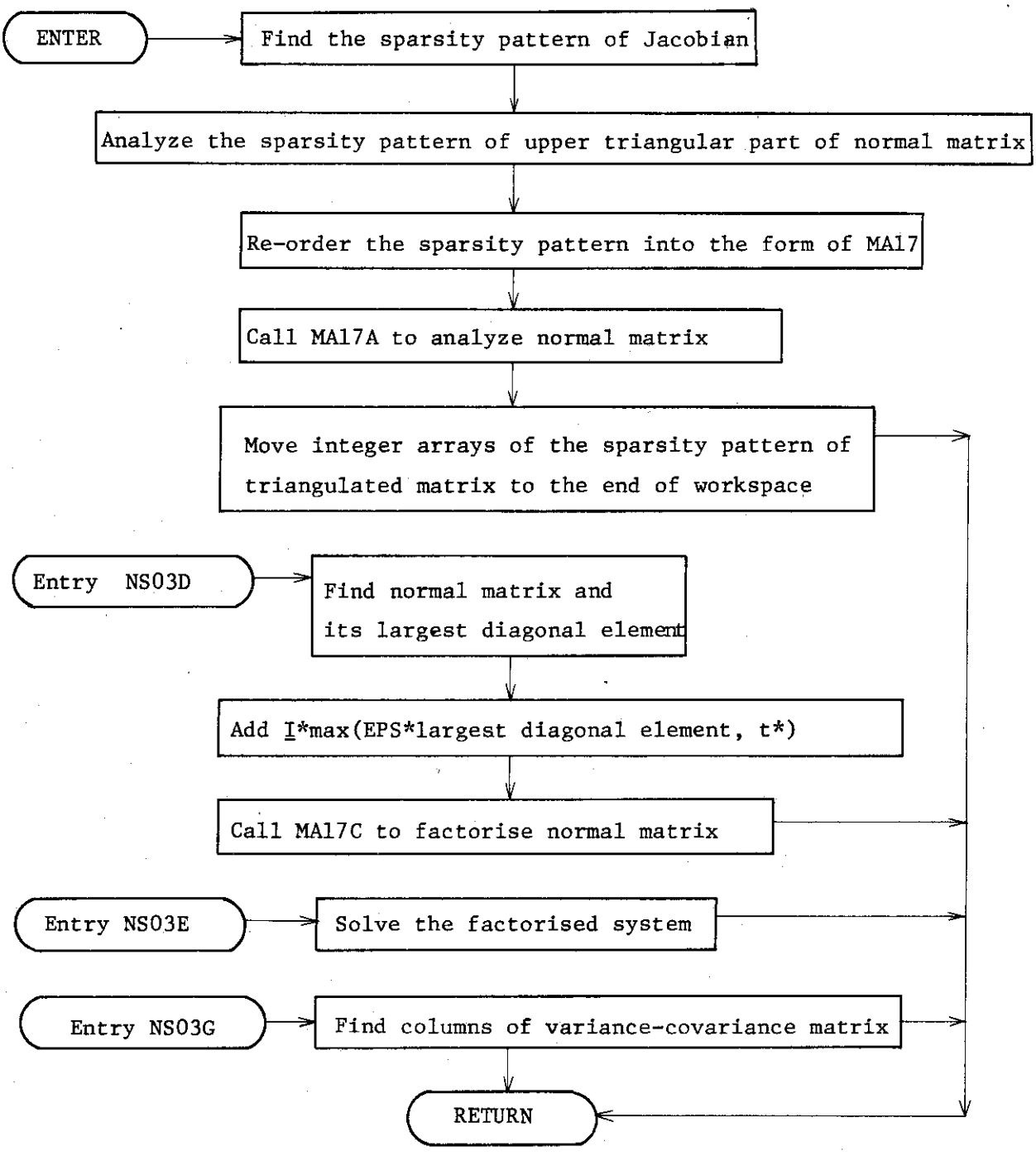

Fig. 4 Flow diagram of NS03C subsidiary to NS03A

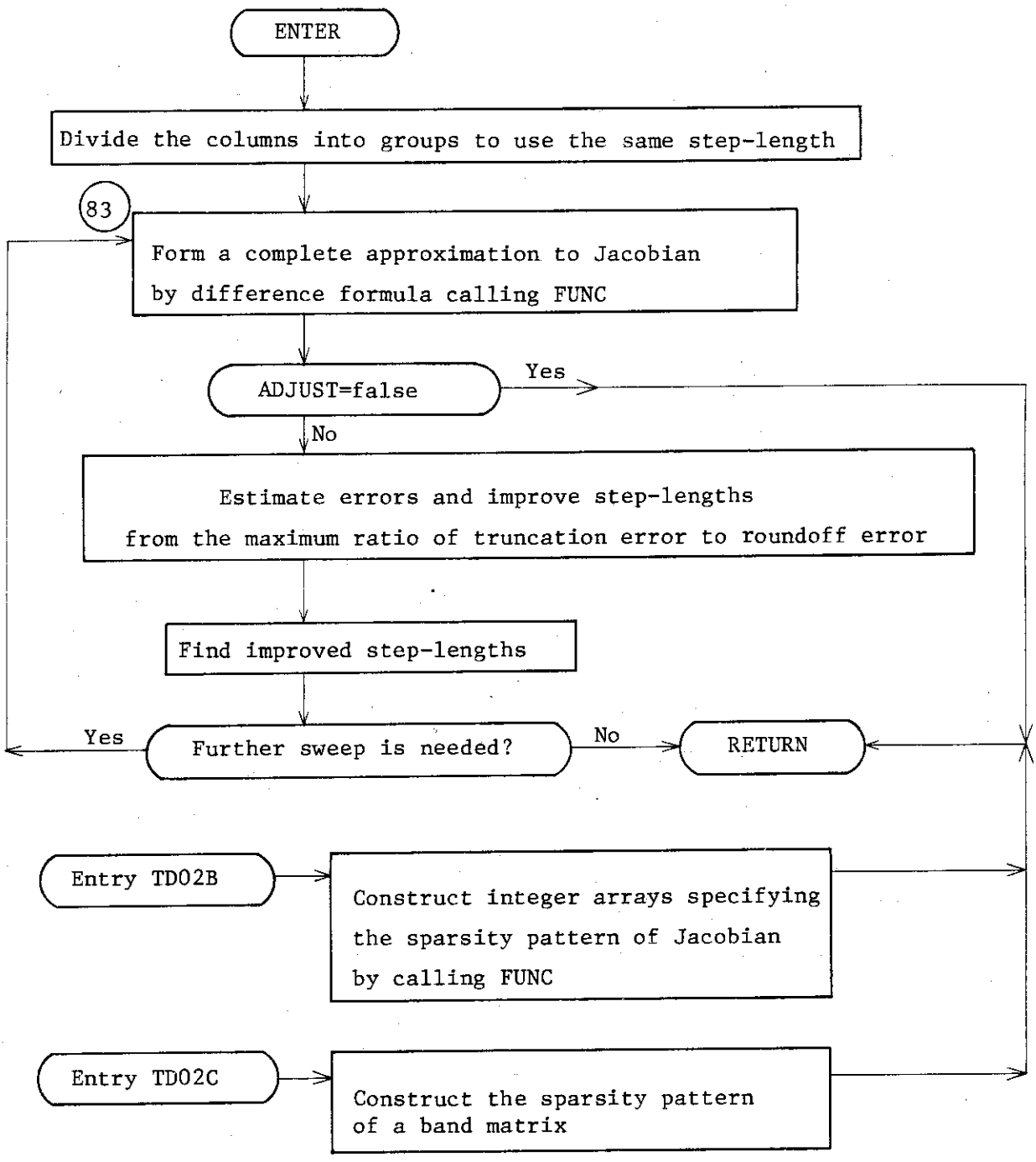

Fig. 5 Flow diagram of TD02A subsidiary to NS03A

数となる。H(N)が差分をとる際のステップの長さで，これはTD02Aで前述のように調整され，その値が出力としてセットされるようになっている。YはFUNCルーチンのためであるが，Fには$\underline{r}(\underline{x})$の値が入れられる。HMAXもNS03Aのと同じだが，もともとはNの大きさの配列で，ステップの長さの上限が入力されるようになっているが，HMAX(1)<0の時にはすべての上限が|HMAX(1)|にとられ，配列の大きさも1でよいようになっている。下限は，上限のEPS1倍にとられる。Aは0でない微分がストアされる配列，IGは(2N+1)以上の大きさの整数配列で，TD02Aに最初に入る時にはIG(1)=0でなければならない。Wは大きさ(M+N)，Zは大きさNの配列で，IAは，AとIRNの配列の大きさを与える。一方，MBDにはバンド構造のヤコビアンのバンドの半幅が入力される。すなわち$|j-k| \geq$ MBDに対して$\partial r_j / \partial x_k = 0$である。

Newton法の変形はBrownも提案している[15]。実変数関数に対し(3)式を1度には1行のみを扱ってTaylor展開により線形化し，ガウスの前方消去法で各行から1つづつ変数を消去している。このようにすると常に最新の情報をアルゴリズムの各ステップで用いることができる。

すなわちi反復過程の第1ステップでは[16]

$$f_1(\underline{x}) = f_1(\underline{x}^i) + \sum_{j=1}^{N} (\partial f_1 / \partial x_j)_i (x_j - x_j^i) \qquad (24)$$

から，$|\partial f_1 / \partial x_j|$が最大になる　例えばj=Nに対し，(24)式が0になるように

$$x_N = x_N^i - \sum_{j=1}^{N-1} [(\partial f_i / \partial x_j) / (\partial f_1 / \partial x_N)]_i (x_j - x_j^i) - f_1(\underline{x}^i) / (\partial f_1 / \partial x_N)_i \qquad (25)$$

次の第2ステップでは(25)式の右辺を$L_N^i(x_1, \cdots, x_{N-1})$とおく時，

$$g_2(x_1, \cdots, x_{N-1}) = f_2(x_1, \cdots, x_{N-1}, L_N(x_1, \cdots, x_{N-1})) \qquad (26)$$

について(24)式と同様に1次までのTaylor展開をとり，偏微分が最大値を与える例えばx_{N-1}について(25)式と同様に解く。このようなステップを繰り返し，第Nステップでは

$$g_N(x_1) = f_N(x_1, L_2, L_3, \cdots, L_N) \qquad (27)$$

からx_1を求めるようになる。このx_1をx_1^{i+1}とし，L_jを逆に解いていくことによりx_j^{i+1}が求められるわけである。偏微分はすべて

$$\partial f_k / \partial x_j = [f_k(\underline{x} + h\underline{e}_j) - f_k(\underline{x})] / h \qquad (28)$$

で近似するが，この際のステップhを，局所的2位収束が保証されるように

$$h_j = \max\{\alpha_{kj}, 5 \times 10^{-\beta+2}\} \qquad (29)$$
$$\alpha_{kj} = \min\{\max(|f_1|, |g_2|, \cdots, |g_k|), 0.001|x_j|\}$$

と選んでいる。ここでβは計算機の有効桁数である。

このアルゴリズムによるルーチンNONLINは，そのFORTRANプログラムがBrownにより発表されているので[16]，これの整備がなされた。そのプログラム・リストは付録4に，計算の流れはFig. 6に示した。このルーチンの呼び出しは以下による。

CALL NONLIN（N, NUMSIG, MAXIT, IPRINT, X, EPS）

ここでNは方程式の数，すなわち変数の数，NUMSIGは求める有効桁数で，MAXITは入力としては反復の上限回数，出力としては要求された反復回数がセットされる。IPRINTはプリント出力のためには1にする。Xには解の推定値が入力され，出力として計算結果の解がセットされる。EPSはすべての関数値の絶対値がこれ以下になったとき反復計算を終える収束判定値であるが，NUMSIGによっても収束判定がなされるので，実際の計算の終了は，これらのいずれかによる。

付録4，あるいはFig. 6から分るように，$\underline{f}(\underline{x})$の形はAUXFCNルーチンで定義されるが，1つづつ$f_k=0$を解いていくので，線形に近いf_kが先に扱われるように順序付けるとよいことが指摘されている。反復あたりのf_kの計算回数は$(N^2+3N)/2$で，Newton法の(N^2+N)（この他にN^2回の偏微分計算なども必要）よりも少ないが，L_kの計算が必要なので，実際にはf_kの計算が複雑な時のみ計算量が減少する。しかしアルゴリズムから分るように局所的に非常に安定した解法となっている。この方法も2位の収束をしており，線形に近い非線形系に対して効率よい方法であることが示されている。

この他に任意の初期値から収束する手法として，Newton法を常微分方程式に対するEuler法と考え，その数値積分を改良する観点で開発された手法がある。最近では，Boggsによる独立変数を無限領域〔0, ∞〕で定義した方法が注目されている。[17)]

今，$\underline{f}(\underline{x})=0$を解くため，$\underline{G}(t,\underline{x})=0$を，$\underline{G}(0,\underline{x})=0$の解が推定値$x_0$で，$t\to\infty$で$G(t,\underline{x})\to\underline{f}(\underline{x})$となるよう以下のように定義する。

$$\underline{G}(t,\underline{x})=\underline{f}(\underline{x})-\exp(-t)\underline{f}(\underline{x}_0)=0 \tag{30}$$

これより

$$d\underline{x}/dt=-\underline{A}^{-1}(\underline{x})\underline{f}(\underline{x}),\quad \underline{x}(0)=\underline{x}_0 \tag{31}$$

の形の初期値問題が得られる。これに対する数値積分のEuler法は

$$\underline{x}_{i+1}=\underline{x}_i+h(d\underline{x}/dt)_i \tag{32}$$

なる反復を与えるが，これでh＝1とおいたのがNewton法となっている。Robbはvan Melleに従い，(31)式を予測子（predictor）のステップとして，修正子（corrector）として

$$\underline{x}_{i+1}=\underline{x}_i+h[\alpha(d\underline{x}/dt)_{i+1}+(1-\alpha)(d\underline{x}/dt)_i] \tag{33}$$

を用いている。[18)] もし$\alpha=0$，h＝1ならば，これもNewton法となるが，$\alpha=1$，$h\ll 1$ならば収束に多くのステップを要するであろうが，収束はより確かとなるであろう。Robbは(33)式を解く代りに，近似的に以下のアルゴリズムを用いた計算プログラムを発表している。

$$\underline{x}^{(0)}_{i+1}=\underline{x}_i+h(d\underline{x}/dt)_i \tag{34}$$

$$\Delta=[h\underline{J}^{-1}\underline{f}(\underline{x}^{(0)}_{i+1})+h(d\underline{x}/dt)_i]/(1+h\alpha) \tag{35}$$

$$\underline{x}_{i+1}=\underline{x}^{(0)}_{i+1}-\alpha\Delta \tag{36}$$

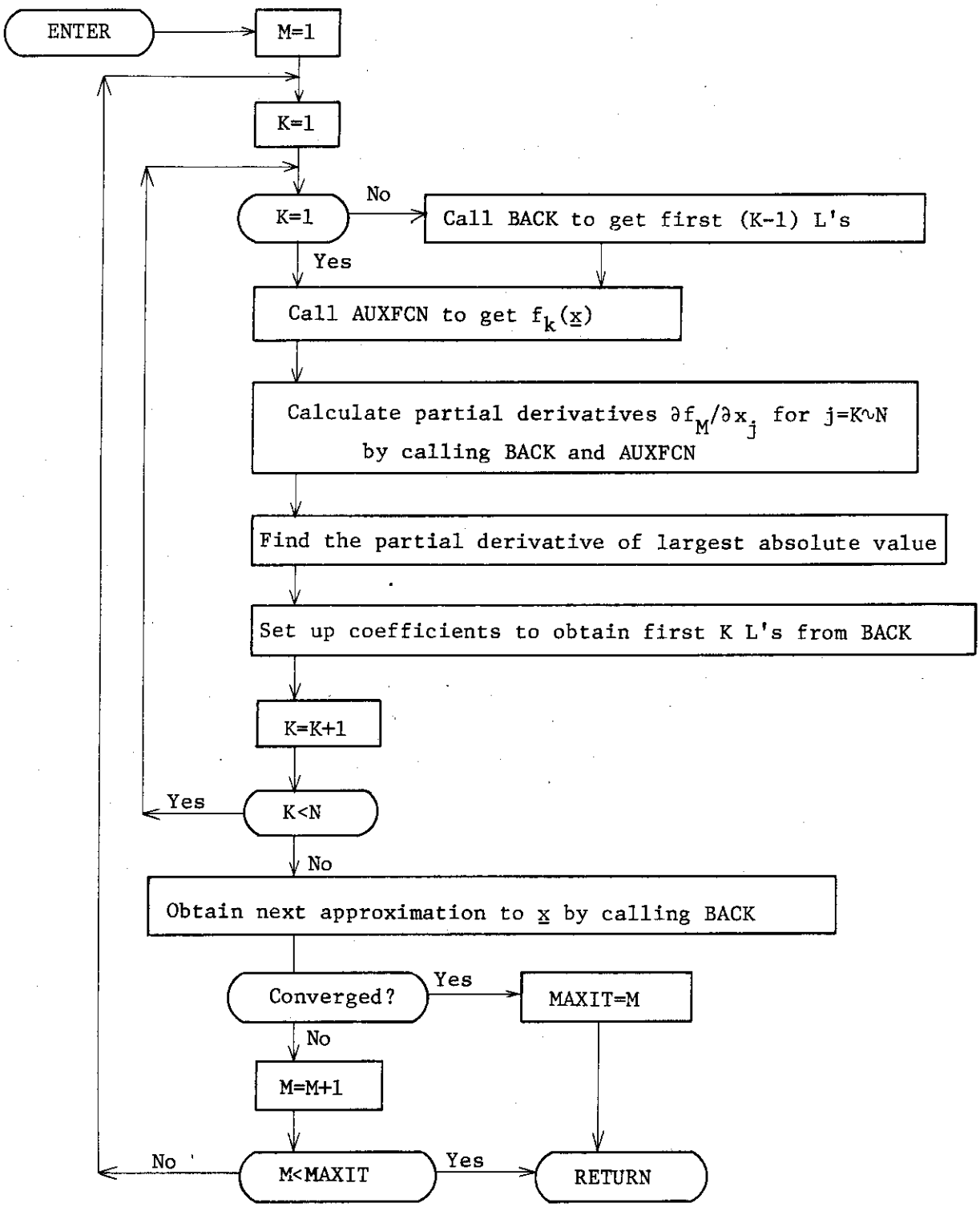

Fig. 6 Flow diagram of NONLIN to solve $\underline{f}(\underline{x})=0$ from an estimate of $\underline{x}$

$$h(d\underline{x}/dt)_{i+1} = h(d\underline{x}/dt)_i - \underline{\Delta} \quad (37)$$

すなわち，(34)が予測子のステップで，(35)～(37)が修正子ステップとなっている。

このルーチンINTECHの計算の流れはFig.7に示したが，計算の効率を上げるため，変数を線形でしか現れないものとそれ以外に分類し，前者に対しては修正子ステップのみを適用している。またヤコビアン行列$\underline{B}$は，非線形の変数の数の5倍のステップ毎にのみ計算している。更に(35)～(37)式のαとhの選択を｜f｜が5％以上減少している限りは$\alpha=0$，$h=1$のNewton法を用いるが，そうでなければ$\alpha=1$，$h=0.01$として計算をやりなおしている。その結果｜f｜が2％以上減少するならば，αは0.8倍とし，hは$(1.7-0.85h+0.15/h)$倍をとるようにしている。ただし｜f｜が100倍以上に増加する時には，$\alpha=1$，$h=\max(h/2, 0.2)$として計算をやりなおし，0.98倍と100倍の間の時には$\alpha=1$，hは$\min(1.3, 0.6/h)$倍して修正子ステップへ進むようになっている。

このルーチンは以下により呼ばれる。

```
CALL INTECH ( Y, YL, DY, PW, DEL, F1, YD, SAVE, YLSV, PWORK, N,
              NY, NL, NSEND )
```

ここでYは非線形の変数の初期値を入力する大きさNYの配列，YLは線形の変数の初期値を入力する大きさNLの配列である。DYは，付録5に例を示したように，DIFFUNルーチンにより計算された関数値が入るN＝NY＋NLの大きさの配列である。PWは同じく付録5に示したMATSETルーチンにより計算されたヤコビアン行列のための(N×N)の2次元配列で，MINVルーチンにより逆行列の値もセットされる。DELはFig.7にも示した関数の絶対値に要求する精度，F1はPWとDYの積のための大きさNの配列，YDはYの$\delta\underline{x}$のための大きさNYの配列，SAVEはYとYDをストアする(2, NY)の2次元配列，YLSVはYLをストアする大きさNLの1次元配列である。PWORKはMINVルーチンのための大きさNの配列だが，MINVルーチンは原典と異なるため実際には使われていない。最後のNSENDは反復回数の上限を与える入力パラメータである。

このINTECHルーチンはすでに述べたように4つの補助ルーチン，DIFFUN，MATSET，MINV，MATMULを用いている。DIFFUN(DY, Y, YL, N, NY, NL)とMATSET(PW, Y, YL, N, NY, NL)の引数は，INTECHのと同じである。MINV(PW, DET, N, NZ, F1, PWORK)はPW(N, N)の逆行列を求めるルーチンで，原典には与えられていなかったので，すでに述べたFACOM SSL[11]の掃き出し法によるMINV2Sルーチンをその中で呼ぶようにした。なお，MATMUL(PW, DY, F1, N)ルーチンでは

$$F1(J) = \sum_{k=1}^{N} PW(J,K) * DY(K), \quad J=1 \sim N \quad (38)$$

を求めている。

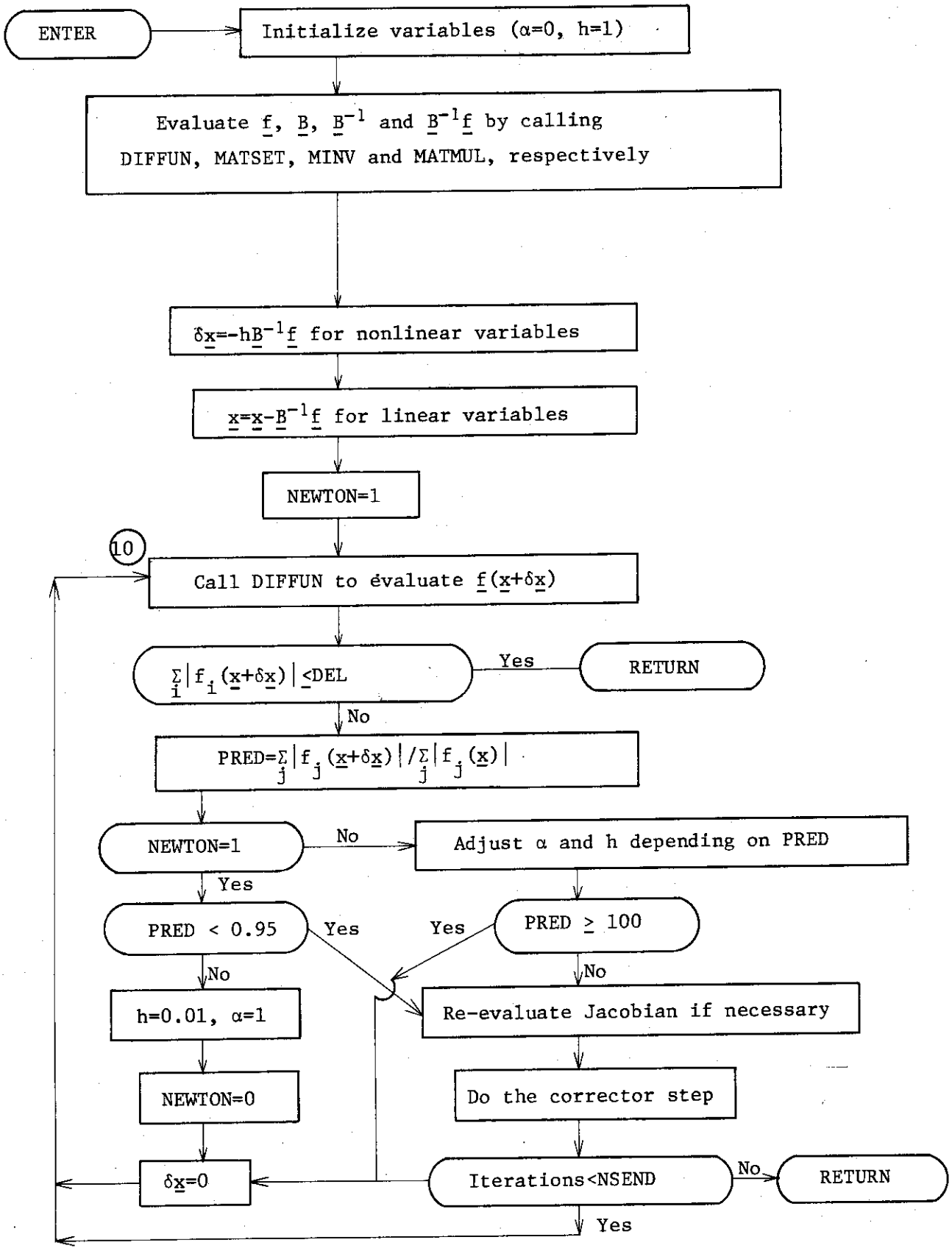

Fig. 7 Flow diagram of INTECH to solve $\underline{f}(\underline{x})=0$ from an estimate of $\underline{x}$

3. 射影法（projection method）ルーチン

射影法は，もともとn次元連立1次方程式

$$\underline{A}\underline{x} = \underline{b} \tag{39}$$

を解く反復法として開発された。1次元射影法は，$\underline{w}_k$ を単位ベクトルとする時，i番目の反復計算では，

$$\underline{x}_{i+1} = \underline{x}_i + dx_i \underline{w}_k$$

のように一時に解ベクトルの1つのk成分のみが補正される。すなわちi番目の反復での残差ベクトル

$$\underline{r}_i = \underline{b} - \underline{A}\underline{x}_i$$

の2次形式$||\underline{r}_i||$を最小にするように，$\underline{A}$のk列目のベクトル$\underline{a}_k$を用い以下のようにdx_i が決められる。

$$dx_i = \underline{r}_i{}^T \underline{a}_k / ||\underline{a}_k|| \tag{40}$$

しかしこの1次元射影法は収束が遅い。2次元射影法では，k番目とj 番目の成分を用い，

$$\alpha_k = \underline{r}_i{}^T \underline{a}_k / ||\underline{a}_k||, \quad \alpha_j = (\underline{r}_i - \alpha_k \underline{a}_k)^T \underline{a}_j / ||\underline{a}_j||,$$

$$dx_{ik} = (\alpha_k - \alpha_j \underline{a}_k{}^T \underline{a}_j / ||\underline{a}_k||) C_{jk},$$

$$dx_{ij} = (\alpha_j - dx_{ik}\ \underline{a}_k{}^T \underline{a}_j)\ C_{jk}, \quad C_{jk} = 1/(1-\cos^2\theta_{jk}),$$

$$\cos\theta_{jk} = \underline{a}_j{}^T \underline{a}_k / (|\underline{a}_j| \cdot |\underline{a}_k|) \tag{41}$$

これより残差ベクトルは以下のように減少する。

$$\underline{r}_{i+1} = \underline{r}_i - dx_{ik}\underline{a}_k - dx_{ij}\underline{a}_j \tag{42}$$

この2次元射影法は1次元より収束が速い。収束に要求される条件は，1次元，2次元とも，行列Aが非特異ということのみである。

Tokko & Keller[19]は，これを拡張して，kの3つの値に対して

$$\underline{r}_{i+1}{}^T \underline{a}_k = 0 \tag{43}$$

を要求する3次元射影法を示し，kの選択は$||\underline{r}_i|| - ||\underline{r}_{i+1}||$を最大にするようにしている。この3次元射影法は2次元のより更に収束が速くなっている。

Harms & Keller[20]は，この3次元射影をk次元へ一般化し，かつGauss-Seidel 法くらいの計算

速度をもつように改良することを考えている。n次元の(39)式を(n+1)次元射影法により，2つの(n−1)次元体系にする。この2つの(n−1)次元の係数行列は，対称正定値の同一の行列である。従ってこの(n−1)次元体系はGauss-Seidel法で収束するし，ガウス消去法も軸選択のない簡単な形で用いることが出来る。もう1つの方法としては，この1次元低減法を繰り返し適用して，(39)を1次元体系にしてしまう直接法を提案している。しかしこの方法は正確な解は与えるが，膨大な計算機容量と演算回数を要求するので，この点の改良が必要であると結論している。

この射影法の非線形系への応用については，まずMacEachern & Kellerの仕事がある。[21] $\underline{f}(\underline{x})=0$を1次元射影法で解いているので，i反復計算のk番目のステップでは

$$\underline{x}_{ik}=(x_1^{i+1},\ \cdots,\ x_k^{i+1},\ x_{k+1}^{i},\ \cdots,\ x_n^{i})$$

となっている。反復

$$\underline{x}_{i,\,k+1}=\underline{x}_{ik}+d\underline{x}_{i,\,k+1}\ ;\quad k=0\sim n-1,\ i=0,\ 1,\ 2,\cdots$$

は，残差ベクトルのノルムの差

$$||\underline{r}(\underline{x}_{ik})||-||\underline{r}(\underline{x}_{i,\,k+1})||$$

が最大になるようにしている。すなわち微分を0にする条件から，アルゴリズムは以下のようになる。①初期近似ベクトル$\underline{x}_{00}$を選ぶ。②近似ベクトルに対する残差ベクトルを計算する。③$\underline{A}_{k+1}(\underline{x}_{ik})$を$\underline{x}_{ik}$でのヤコビアン行列の(k+1)番目の列ベクトルとし，以下の計算をする。

$$d\underline{x}_{i,\,k+1}=-\underline{r}(\underline{x}_{ik})^{T}\underline{A}_{k+1}(\underline{x}_{ik})/||\underline{A}_{k+1}(\underline{x}_{ik})||,$$

$$\underline{x}_{i,\,k+1}=\underline{x}_{ik}+d\underline{x}_{i,\,k+1} \qquad (44)$$

④収束の判定をし，収束していなければ，次のkに対して②から④までを繰返す。これのFORTRANプログラムと例題は，Grorg & Kellerによって報告されている。[22] 又Schmitzら[23]は1～3次元射影法による，線形，および非線形システムのためのAlgolプログラムのリストを示し，詳細な説明も付けている。

各ステップで近似ベクトルのk個の成分が修正される，k次元の非線形射影法と，それの収束については，Georg & Kellerがまとめている。[24] この際には残差ベクトルが，前の近似解を使って計算されたヤコビアン行列の，k個の列ベクトルに直交するようにされる。従ってこのk次元部分空間への残差ベクトルの射影の順序が重要となる。各反復毎に同じ順序で射影していく方法を定常的射影法と呼び，これのFORTRANプログラムPROJAのリストが記されている。まだ最適なアルゴリズム確立まではいっていないが，数値例からNewton法などと同様に用いることが出来ることが示されている。このPROJAルーチンは，すでに当核設計研究室で整備されJSSLに登録されている。[14]

このルーチンの計算の流れはFig. 8に示したが，[24] これの呼び出しは，付録6に示したように以下による。[14]

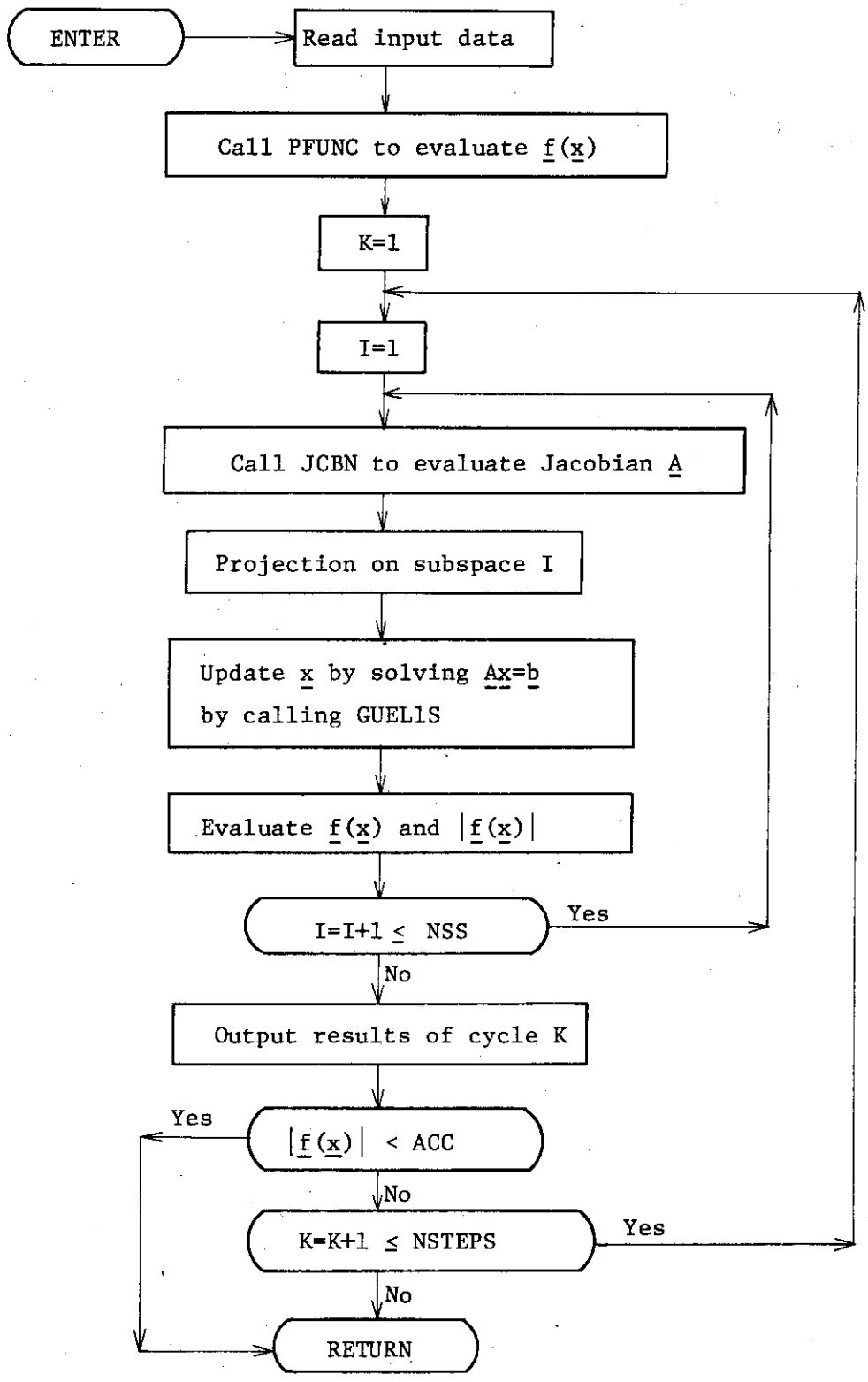

Fig. 8 Flow diagram of PROJA to solve $\underline{f}(\underline{x})=0$ from an estimate of $\underline{x}$

CALL PROJA(NDIM, NCOL, A, B, F, JFX, X, NMAX)

ここで入力はNMAXのみで，次元数以上の整数値を入れればよい。残りの仮引数はすべて作業領域で，NDIM(NMAX)，NCOL(NMAX, NMAX)が整数型配列で，A(NMAX, NMAX)，B(NMAX)，F(NMAX)，JFX(NMAX, NMAX)，X(NMAX)が実数型配列である。ただしPROJAルーチンには入力カードが必要で，1枚目には(3I5)で次元数，反復計算回数(サイクル数)の上限，各反復過程での射影の回数が入力される。2枚目には(F10.5)で収束判定精度ACCが入力され，Fig.8にも示したように$|\underline{f}(\underline{x})| <$ ACCで判定がなされる。3枚目のカードには(及び必要なときには次のカードにも続けて)(8F10.5)で解ベクトルの近似値が入力される。次いで，まず最初に射影する部分空間の次元数が(I5)で入力され，その次のカードにはその部分空間に属する次元(ヤコビアンの列)を(16I5)で次元数だけ入力する。この射影部分空間に関する1組のカードが，1枚目のカードに入力した各反復過程での射影回数だけ繰り返えされる。

なお，$\underline{f}(\underline{x})$の形は付録6に示したようにPFUNC(X, F, N)ルーチンで定義しなければならない。ここでXが変数，Fが関数で，それぞれNMAX(定数)の1次元実数型配列である。またヤコビアンは，JCBN(JFX, X, N)ルーチンで，大きさ(NMAX, NMAX)(これも定数で与える)の実数型配列JFX(J, K)に$\partial f_j / \partial x_k$を定義しなければならない。なお線形系の式を解くGUELIS[14]は藤村によりJSSLとして整備された新しいルーチンである。

なおGeorg & Kellerは，最近，k次元部分空間への残差ベクトルの射影の順序を，ヤコビアンの列ベクトルと残差ベクトルとの間の角に基づいてdynamicalに選んでいくアルゴリズムを発表している[25]。また，Nguyen & Kellerは，各反復ステップでヤコビアンと残差を再計算するのでなく，$\underline{x}$の全成分が1通り修正されるまでは再計算せずに使っていくprojection-based法を開発し，普通の射影法より計算時間が短く，計算のサイクル数も少なくなることを示している[26]。

4. ベンチマーク・テストと検討

前章までで連立非線形方程式の解法ルーチンとして，準Newton法のNS01A[9]，NS03A[12]，NONLIN[16]，INTECH[18]，及び射影法のPROJA[24]の5つのルーチンについて説明した。本章では，これらルーチンのベンチマーク・テスト結果について述べる。

テスト問題としては，すでに引用した文献中で用いられている9つを採用した。これらをTable 1にまとめた。例題1から7まではRobb[18]が採用しているが，例題3はBroyden[6]，Shampine & Gordon[7]も用いている。例題4は，初期値と解の間の点でヤコビアン行列が特異となるのが特徴である。例題5はf_1とf_2の係数の大きさの差(scaling)の関係で困難な問題とされており，Powell[9]，Brown[16]も採用している。次の例題6は，よく知らされているRosenbrockの方程式で，Powell[9]も採用している。Freudenstein & Rothの方程式である例題7も，むつかしい問題としてPowellも引用している。例題8はGrorg & Kellerのもので[24]，最後の4次元系の例題9はBrown[16]から採用した。

まずTable 2に準Newton法の4つの解法ルーチンにより求められた解を，FACOM 230/75

CALL PROJA(NDIM, NCOL, A, B, F, JFX, X, NMAX)

ここで入力はNMAXのみで，次元数以上の整数値を入れればよい。残りの仮引数はすべて作業領域で，NDIM(NMAX)，NCOL(NMAX, NMAX)が整数型配列で，A(NMAX, NMAX)，B(NMAX)，F(NMAX)，JFX(NMAX, NMAX)，X(NMAX)が実数型配列である。ただしPROJAルーチンには入力カードが必要で，1枚目には(3I5)で次元数，反復計算回数(サイクル数)の上限，各反復過程での射影の回数が入力される。2枚目には(F10.5)で収束判定精度ACCが入力され，Fig.8にも示したように$|\underline{f}(\underline{x})| <$ ACCで判定がなされる。3枚目のカードには(及び必要なときには次のカードにも続けて)(8F10.5)で解ベクトルの近似値が入力される。次いで，まず最初に射影する部分空間の次元数が(I5)で入力され，その次のカードにはその部分空間に属する次元(ヤコビアンの列)を(16I5)で次元数だけ入力する。この射影部分空間に関する1組のカードが，1枚目のカードに入力した各反復過程での射影回数だけ繰り返えされる。

なお，$\underline{f}(\underline{x})$の形は付録6に示したようにPFUNC(X, F, N)ルーチンで定義しなければならない。ここでXが変数，Fが関数で，それぞれNMAX(定数)の1次元実数型配列である。またヤコビアンは，JCBN(JFX, X, N)ルーチンで，大きさ(NMAX, NMAX)(これも定数で与える)の実数型配列JFX(J, K)に$\partial f_j / \partial x_k$を定義しなければならない。なお線形系の式を解くGUELIS[14]は藤村によりJSSLとして整備された新しいルーチンである。

なおGeorg & Kellerは，最近，k次元部分空間への残差ベクトルの射影の順序を，ヤコビアンの列ベクトルと残差ベクトルとの間の角に基づいてdynamicalに選んでいくアルゴリズムを発表している[25]。また，Nguyen & Kellerは，各反復ステップでヤコビアンと残差を再計算するのでなく，$\underline{x}$の全成分が1通り修正されるまでは再計算せずに使っていくprojection-based法を開発し，普通の射影法より計算時間が短く，計算のサイクル数も少なくなることを示している[26]。

4. ベンチマーク・テストと検討

前章までで連立非線形方程式の解法ルーチンとして，準Newton法のNS01A[9]，NS03A[12]，NONLIN[16]，INTECH[18]，及び射影法のPROJA[24]の5つのルーチンについて説明した。本章では，これらルーチンのベンチマーク・テスト結果について述べる。

テスト問題としては，すでに引用した文献中で用いられている9つを採用した。これらをTable 1にまとめた。例題1から7まではRobb[18]が採用しているが，例題3はBroyden[6]，Shampine & Gordon[7]も用いている。例題4は，初期値と解の間の点でヤコビアン行列が特異となるのが特徴である。例題5はf_1とf_2の係数の大きさの差(scaling)の関係で困難な問題とされており，Powell[9]，Brown[16]も採用している。次の例題6は，よく知らされているRosenbrockの方程式で，Powell[9]も採用している。Freudenstein & Rothの方程式である例題7も，むつかしい問題としてPowellも引用している。例題8はGrorg & Kellerのもので[24]，最後の4次元系の例題9はBrownから採用した[16]。

まずTable 2に準Newton法の4つの解法ルーチンにより求められた解を，FACOM 230/75

Table 1 Test problems of nonlinear simultaneous equations and their solutions

Example	$f_i(x)$, i=1,2 (or i=1∿4)	Initial estimate	Solution
No. 1	$f_1 = 4+x_1+x_2-x_1^2+2x_1x_2+3x_2^2$, $f_2 = 1+2x_1-3x_2+x_1^2+x_1x_2-2x_2^2$	(-2.057, -7.503) (0,1)	∿(3.339, -2.984) ∿(-1.5334,0.061121)
No. 2	$f_1 = x_1^2-x_2+1$, $f_2 = x_1-\cos(\frac{\pi}{2}x_2)$	(1,0), (-1,1)	(0,1), $(-1/\sqrt{2}, 1.5)$
No. 3	$f_1 = [\sin(x_1x_2)-x_2/(2\pi)-x_1]/2$ $f_2 = [1-1/(4\pi)][\exp(2x_1)-e]+ex_2/\pi-2ex_1$	(0.4, 3.0)	∿(0.3000, 2.838)
No. 4	$f_1 = x_1$, $f_2 = 10x_1/(x_1+0.1)+2x_2^2$	(3,1)	(0,0)
No. 5	$f_1 = 10,000x_1x_2-1$, $f_2 = \exp(-x_1)+\exp(-x_2)-1.0001$	(0,1)	$\sim(1.098\times10^{-5}, 9.106)$
No. 6	$f_1 = 10(x_2-x_1^2)$, $f_2 = 1-x_1$	(-1.2, 1.0)	(1, 1)
No. 7	$f_1 = x_1(x_1(5-x_1)-2)+x_2-13$, $f_2 = x_1(x_1(1+x_1)-14)+x_2-29$	(15, -2)	(4, 5)
No. 8	$f_1 = x_1^2+x_2^2-4$, $f_2 = x_1^2-x_2^2$	(2, 3)	$(\sqrt{2}, \sqrt{2})$
No. 9	$f_1 = x_4x_1/3+x_4x_2/6-x_4^3/12$, $f_2 = x_4x_1/6+x_2/3+(1-x_4)x_3/6-(x_4^2+x_4+1)/12$, $f_3 = (1-x_4)x_2/6+(1-x_4)x_3/3+(x_4^3+x_4^2+x_4-3)/12$, $f_4 = 3(x_3-x_1)x_4^2+2(x_3-x_2)x_4+x_3-x_2+2(x_1^2-x_3^2)+2x_2(x_1-x_3)$	(0, 0.01, 1, 0.75)	$\begin{pmatrix} -0.041666\ldots, \\ 0.208333\ldots, \\ 0.958333\ldots, \\ 0.5 \end{pmatrix}$

Table 2 Computation times (CPU msec) and solutions (in parentheses) for 4 subroutines of quasi-Newton methods

Example	NS01A	NS03A		NONLIN	INTECH
		HMAX=1	HMAX=0		
No.1a	64(3.3386, -2.9844)	124(3.3388, -2.9845)	53(3.3386, -2.9844)	13(-1.5334, 0.061121)	28(3.3386, -2.9844)
No.1b	48(-1.5335, 0.061113)	74(-1.5335, 0.061109)	32(-1.5334, 0.061121)	7(-1.5334, 0.061121)	23(-1.5334, 0.061121)
No.2a	34(0.000359, 0.99977)	43(0.001381, 0.99913)	39(0.000973, 0.99938)	8(-0.70710, 1.49999)	32(0.000781, 0.99950)
No.2b	27(-0.70757, 1.50020)	23(-0.70688, 1.49965)	21(-0.70711, 1.50000)	5(-0.70695, 1.49911)	8(-0.70702, 1.4999)
No.3	38(0.30002, 2.8377)	34(0.29923, 2.8369)	23(0.30071, 2.8401)	5(0.29917, 2.8363)*	29(0.29926, 2.8368)
No.4	241(-0.000447, -0.1508)	480(-0.000947, -0.2188)	49(-1.25×10^{-9}, -0.01439)	8(-5.9×10^{-54}, 0.01566)	65(-2.4×10^{-6}, -0.3473)**
No.5	223(1.462×10^{-5}, 6.8398)	475(1.469×10^{-5}, 6.8057)	203(1.467×10^{-5}, 6.8174)	12(1.099×10^{-5}, 9.0966)	52(1.105×10^{-5}, 9.0386)
No.6	101(1.00000, 1.00003)	179(1.00000, 0.99990)	124(1.00000, 1.00000)	4(1.00002, 1.00012)	21(1.0000, 1.0000)
No.7	93(4.0000, 5.0000)	64(4.0000, 5.0000)	42(4.0000, 5.0000)	9(4.0000, 5.0000)	31(4.0000, 5.0000)
No.8	32(1.41421, 1.41422)	40(1.41417, 1.41425)	27(1.41421, 1.41421)	6(1.41421, 1.41421)	12(1.4142, 1.4143)
No.9	48(-0.042107, 0.21091, 0.95933, 0.50277)	213(-0.057033, 0.26260, 0.97312, 0.55033)	812(-0.087436, 0.38886, 1.01531, 0.66177)***	15(-0.042187, 0.20896, 0.95843, 0.50055)	117(-0.042600, 0.21207, 0.95927, 0.50373)

* Converged by reordering the equations, ** Converged only with DEL=0.25, *** Not converged even with 100 calculations of $\underline{r}(\underline{x})$.

によるCPU計算時間と共にまとめておいた。NS01Aについては，付録1に示した例題7に対するテスト・プログラムから分るように，微分を数値的に求めるためのステップSTEPは0.01，$||\underline{f}||$に要求する精度ACCは10^{-6}，$\underline{f}(\underline{x})$の計算回数上限MAXFUNは100，$|\delta\underline{x}|$の上限DMAXは10と，Powellと同様にとった[9]。ただし例題4は，DMAX＝10では計算が打ち切られるので30とした。まず例題1については，初期値が(−2.057，−7.503)の1aも，(0,1)の1bも特に問題なく，1aの場合は$\underline{f}(\underline{x})$の計算回数は18，1bに対しては14回で収束している。例題2も初期値が(1,0)の2aに対して10回，(−1,1)の2bに対して8回の計算で解が求められており，例題3も11回の計算で収束している。なおRobbは例題3の解としてx_2〜2.8を与えているが，〜2.838の方が正しい。例題4は上述のように，DMAX＝10では，$\underline{f}(\underline{x})$の勾配が小さくなり解をDMAXより遠方にもっていくようになるのでエラーとして計算が打ち切られる。DMAX=30の際の解への収束状態をFig.9に示したが，12，13回目の計算で$\underline{x}$=(2.8，−0.08)近傍に来るが，それから非常にゆっくりと，特に$|x_1|$の値が小さくなり，54回目で$||\underline{f}||$が3×10^{-6}となる。しかしx_1が−0.1近傍になることもあり，$||\underline{f}||$が10^{-6}以下になるのは69回目である。正解は(0，0)だが，x_2＝−0.15とあまい結果を与えている。次の例題5も困難な問題で，15回目くらいでx_1は10^{-5}のオーダになるが，x_2は3程度で，それから徐々に増加して61回目で収束しているが，正解は求められていない。Powellは[9]，STEP＝0.001，DMAX＝20，ACC＝10^{-10}で223回目で収束し，(1.106×10^{-5}，9.038)を得ている。この場合でも77回目で$\underline{x}$=(1.457×10^{-5}，6.862)，$||\underline{f}||=8.7\times10^{-7}$となっている。(18)式で述べたように微分を求める際のステップがすべての変数に対して同一なので，このようなscalingの点で問題のある方程式の正解を求めるには，$||\underline{f}||$に要求する精度を上げる必要があろう。例題6はFig.10に示したように，大体$x_2=x_1^2$に沿って，15回目くらいで$\underline{x}$=(0.34，0.09)に到達し，それから徐々に(1,1)に近づき28回目で収束している。例題7も初期値が解から相当離れていて，Fig.11に示したようにx_1がまず4近くになるが，x_2は−14くらいで，それが徐々に5に近づいていくため，26回目でようやく収束をみている。例題8，9は，両者とも10回目で収束しており，一応問題ないようである。

次にヤコビアンが疎の場合に対するNS03Aによる結果だが，Table 2には，HMAX(1)＝1としてNS01Aの際同様，差分によりヤコビアンを求めた場合と，HMAX(1)＝0として，与えた微分形により求めた場合の両者の結果を示してある。付録2あるいは3の例題5に対するテスト・プログラムから分るように，$||\underline{f}||$に要求する精度はNS01Aと同様SAC＝10^{-6}とし，$\underline{f}(\underline{x})$の計算回数上限MAXFUNも100ととっている。一方$|\delta\underline{x}|$に対する要求精度STPMINは0.0とおいたので，内部的な$|\delta\underline{x}|<10^{-14}|\underline{x}|$のみが要求されるが，$||\underline{f}||<10^{-6}$の方が一般には先に満足されるので，この要求は一般には使われない。従って収束条件はNS01Aと同じと考えてよい。しかし倍精度ルーチンとなっている点が他とは異なっている。

例題1は，ヤコビアンにも(23)式の$\underline{A}$にも0の要素はないので，特に疎である特徴は使えていない。(19)式のt^*の計算のため，t^*が0のまゝで収束する場合を除いて$\underline{r}(\underline{x})$を計算する回数が反復回数より多く，HMAX＝1の時は，例題1aでは22反復で28回の計算，1bでは13反復で15回の計算を要求し，いずれも計算時間はNS01Aの2倍近くになっている。しかしHMAX＝0の時には，1aで9反復で10回計算，1bではt^*＝0のまゝで6反復で収束しており，計算時間もNS01Aより2〜3割短くなっている。例題2はヤコビアンも$\underline{A}$も，4要素のうち2要素が0で，

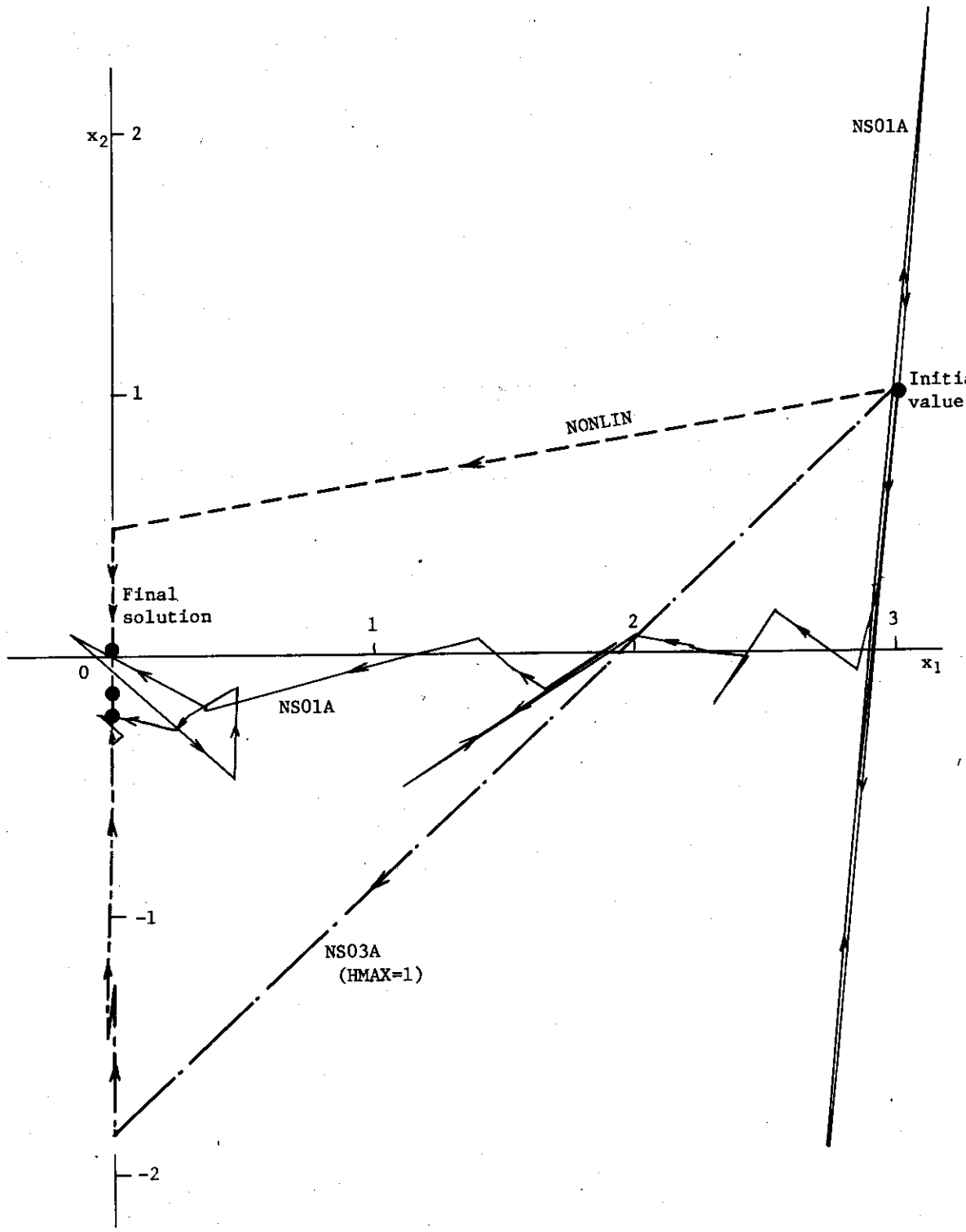

Fig. 9 Trajectories of searching a solution for Example No.4 by NS01A, NS03A (HMAX=1) and NONLIN

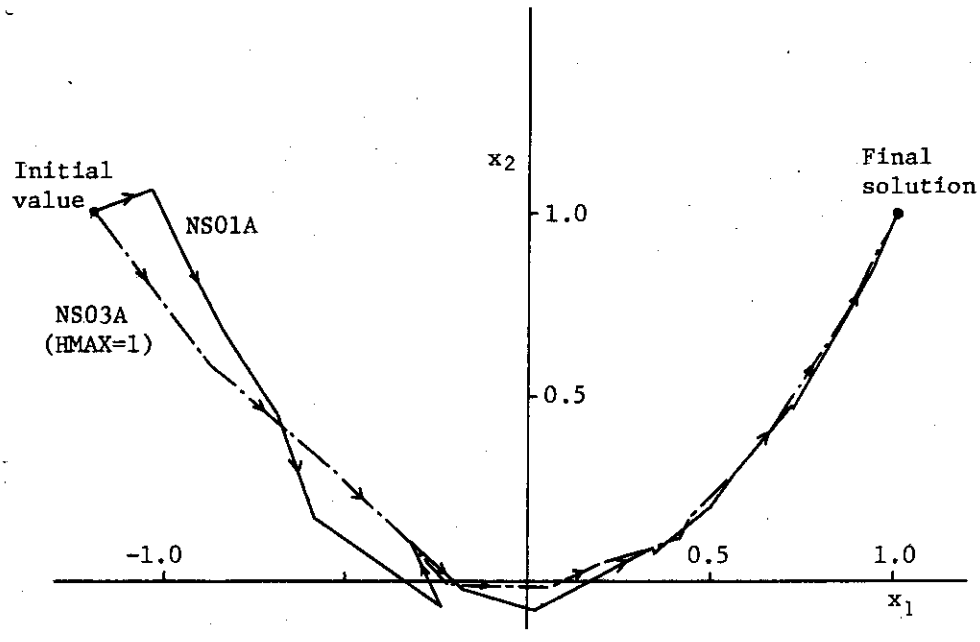

Fig. 10 Trajectories of searching a solution for Example No.6 by NS01A and NS03A (HMAX=1)

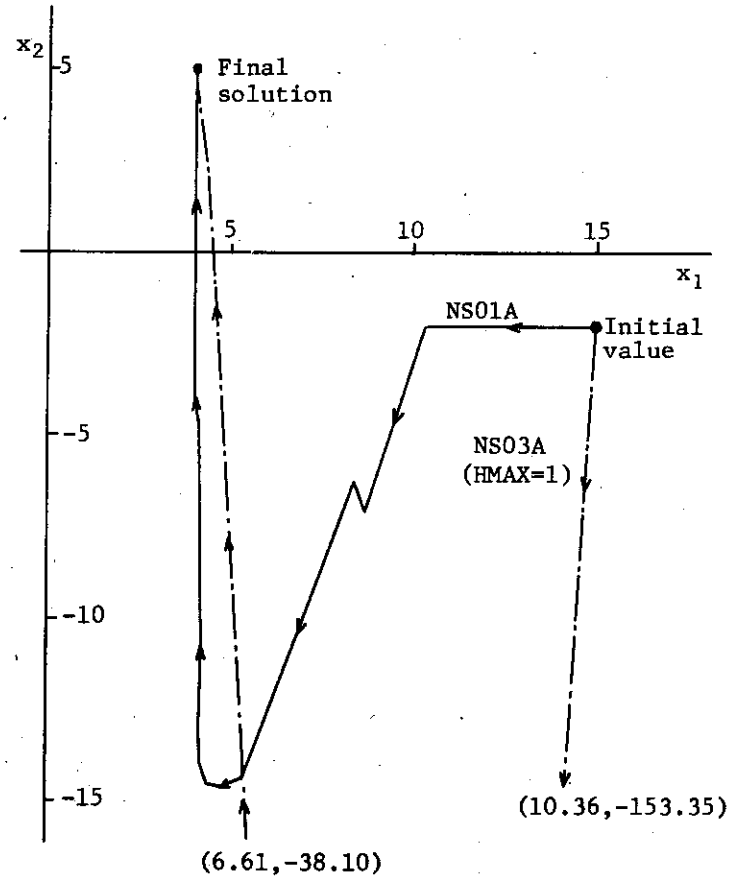

Fig. 11 Trajectories of searching a solution for Example No.7 by NS01A and NS03A (HMAX=1)

このためHMAX＝1の際でも，2aでは8反復で10回計算，2bでは4反復で5回計算で収束し，計算時間もNS01Aと殆んど同じであるが，精度はいくらか甘いようである。HMAX＝0では，2aに対して7反復の7回計算，2bに対してはt*＝0のまゝで4反復で収束しているが，計算時間はHMAX＝1の際と比べ殆んど短縮されていない。例題3についてはヤコビアンの1要素のみが0だが，HMAX＝1では6反復の8回計算で，NS01Aより短時間で収束しており，HMAX＝0では4反復の5回計算で，NS01Aの殆んど半分の時間で収束している。例題4はヤコビアンの2要素，Aの3要素が0で，HMAX＝0の際にはt*＝0のまゝで9反復で収束し，NS01Aの1/5の時間で，より精度の高い結果が得られている。しかしHMAX＝1の際には，85反復の92回計算で，NS01Aの2倍の時間を要している。これは，Fig.9に示したように，10反復目で（−0.03，−1.4）付近に来てから後，x_2の値が非常にゆっくりと0に近づいていくためである。

次の例題5はヤコビアンには0の要素はないが，Aはすべて0要素である。しかしNS01Aより収束は遅く，15反復目で（6×10^{-5}，1.7）程度になった後，徐々にx_1は減少し，x_2は増加していくが，結局84反復，92回計算で，NS01Aの2倍の時間を要し，NS01A同様，正解からずれて収束している。HMAX＝0では37反復，38回計算で，NS01Aと同様な結果を得ている。例題6はヤコビアンは3要素が0，Aは2要素が0であるが，HMAX＝1の際には，Fig.10に示したように，NS01Aの際同様，大体$x_2=x_1^2$に沿って動くため，31反復，32回計算で収束し，計算時間はNS01Aの2倍近くになっている。HMAX＝0の際にも同様で，Fig.12に収束状態を示したように，22反復，27回計算で収束しているが，NS01Aより長い計算時間を要している。例題7は，Aには0要素がないが，ヤコビアンの2要素が0で，HMAX＝1でもNS-01Aより短時間で収束している。Fig.11に示したように，x_2が−100程度にまず下がるが，すぐに5反復目で(5.5，−15)付近に戻り，11反復，12回計算で収束している。HMAX＝0の際も同様だが，t*＝0のまゝで8反復目で収束し，計算時間もNS01Aの半分近くである。例題8はヤコビアンには0要素がないが，Aは全部0で，HMAX＝1の際には7反復，9回計算で，HMAX＝0の際にはt*＝0のまゝで5反復で，NS01Aと同程度の計算時間で収束している。これに対し，例題9は，ヤコビアンは16要素のうち3つが0，Aには8つの0要素があるが，HMAX＝1では27反復，35回計算を要求して，計算時間がNS01Aの4倍にもなっている上，精度も良くない。更に，HMAX＝0の際には，98反復目でMAXFUN＝100に到達したため，$||\underline{f}||$はまだ1.0×10^{-5}だがエラーとして計算が打ち切られ，計算時間もすでにHMAX＝1の際の4倍になっている。MAXFUN＝200で198回反復しても，$||\underline{f}||$は9×10^{-6}になるだけで，非常に収束が遅い。

次のBrownによるNONLINの結果であるが，これも要求精度としては，付録4の例題3に対するテスト・プログラムからも分るように，NS01A，NS03Aの$||\underline{f}||<10^{-6}$と同様，すべての$|f_k|<EPS=10^{-3}$ととった。求める有効数字の数NUMSIGは6としたが，すべての例題について6桁の精度になる前にEPSの条件で収束している。なお反復の上限回数MAXITは100ととった。Table 2から分るように，NONLINはすべての例題に対して最少の計算時間で収束している。例題1については，1aは13回反復，1bは6回反復で，初期値に依らず同一の解に収束している。例題2も，2aは7回反復，2bは3回反復で同一の解に収束しており，このようにNONLINは初期値に余り依存しないことが分る。

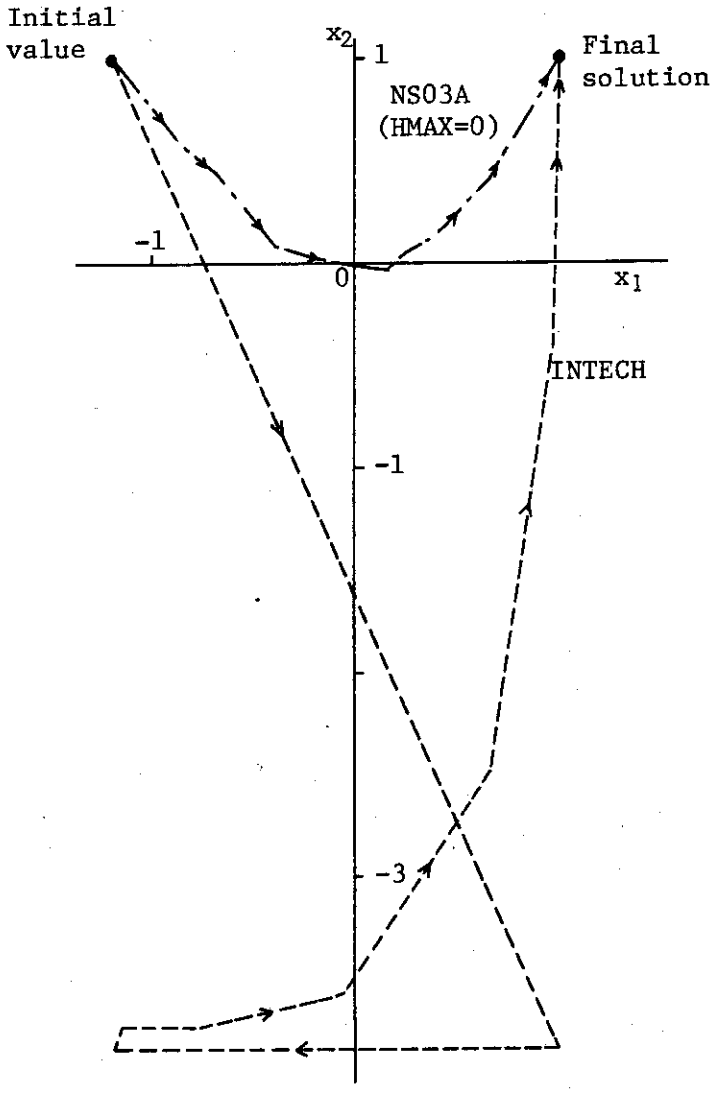

Fig. 12 Trajectories of searching a solution for Example No.6 by NS03A (HMAX=0) and INTECH

第2章で述べたように，このBrownの方法では，線形に近いf_kが先に扱われるようにkを順序付けるとよい。例題3の場合，Table 1に示したf_1とf_2のどちらが線形に近いか一概に言いがたいが，Table 1のままでは100回の反復でも収束の傾向がみられない。これに対し，f_1とf_2を付録4に示したように交換すると，たった4回の反復で収束している。例題4に対しては，Fig. 9に示したように，最初の反復で$(2.1\times10^{-5}, 0.5)$となり，全部で7回の反復で収束しており，他の方法と比べ格段の差がみられる。例題5についても同様で，最初の反復で$(1\times10^{-4}, 2.0)$となり，11反復目で，NS01A，NS03Aでは求められなかった正解に収束しており，この例題がscalingの関係で困難な問題であると言われているのがうそのようである。例題6についても，他の手法ではFig. 10に示したように$x_2 = x_1^2$に沿うため収束が遅いが，NONLINでは，最初の反復でいきなり(1.000，−3.837)となり，その次の反復で収束してしまっている。例題7は最初の反復で(9.34，−688)となり，次いで(6.76，−29.0)，(4.77，−35.6)，(4.93，21.0)を経て(4.22，2.81)となってから，全部で9反復目で収束しており，NS03Aの際と同様，$|\delta \underline{x}|$が非常に長いのが目立つ。例題8は5反復目で，例題9も6反復目で収束しており，NONLINの有用性を顕示している。

次のINTECHに対しては，付録5の例題1aに対するテスト・プログラムからも分るように，収束判定は，$\sum_k |f_k| < \mathrm{DEL} = 10^{-3}$と，他と同程度にとり，反復回数の上限NSENDも100と同様にとった。ただしこのルーチンには，ヤコビアンの微分形を与える必要があり，又主な変数は倍精度になっているので，NS03AルーチンのHMAX＝0に対応するが，線形でしか現われない変数を別個に扱うのが特徴である。しかし例題6と7以外には，線形変数は出てこない。まず例題1であるが，1a，1bともヤコビアンの計算は4回行われ，1aは15回反復，1bは13回反復で収束している。計算時間はNS03A（HMAX＝0）の半分近くである。例題2aは，18回反復，4回ヤコビアン計算で，NS03Aと同程度の時間で収束しているが，2bの方は，Newton法の4反復のみで，2回のヤコビアン計算で，NS03Aの半分以下の時間で収束している。例題3は，変形Newton法も用いて16反復，4回のヤコビアン計算で，NS03Aよりも長い計算時間を要している。

例題4については，まずFig. 9に示したNS01Aと同様な状態で8反復目には(2.3，0.2)付近に来るが，それからは(25，−255)とか(−18615，2.5)などと振動し，Table 2に示したように，ヤコビアンを5回計算した後の41反復目で$\sum_k |f_k| = 0.24$までに到達する。しかしその後は，また，例えば(−108510，1110)とか(3.5，−204)などとなり収束しない。これに対し，例題5の方は29反復，6回ヤコビアン計算で正解に収束しており，計算時間もNS03Aの1/4近くである。例題6は，x_2が線形にのみ現われており，11反復，2回のヤコビアン計算で，NS01Aの1/6の時間で収束している。この収束状態を図示したのがFig. 12で，NS03Aとは異なり，特に$x_2 = x_1^2$に沿っていない。例題7もx_2が線形にのみ現われており，Newton法のみの19反復，4回のヤコビアン計算で，NS03Aより短時間で収束している。収束過程は，NS03AのHMAX＝0と同様，Fig. 11に示したNS03A（HMAX＝1）と大体同じである。次の例題8, 9には線形のみに現われる変数はないが，例題8はNewton法の7反復，2回のヤコビアン計算で，NS03Aの半分以下の計算時間で収束している。例題9も，これはNewton法のみではないが，全部で53反復，11回ヤコビアン計算でNS03Aの1/7程度の計算時間で収束に到っている。

以上で準Newton法のルーチンによる結果を一通り見てきたが，ヤコビアンの微分形を与えなくてよいNS01A，NS03A（HMAX＝1），NONLINの3つのルーチンについては，NONLINが，線形に近いf_kを先に扱うように順序付ける必要はあるが，すべての場合に最少の計算時間で，初期値に余り依存しなく収束しており，特に困難な問題には有用である。NS03Aは(3)式の$\underline{r}(\underline{x})$のヤコビアン，及び$\underline{A}$が疎の場合の倍精度ルーチンであるので，これらが疎の問題はNS01Aよりうまく解ける場合もあるが，大体においてNS01Aと同程度の性能をもっていると考えてよいだろう。ヤコビアンの微分形を与えなくてはならない2つのルーチン，NS03A（HMAX＝0）とINTECHについては，後者も主な変数は倍精度になっているのでNS03Aと同様だが，線形変数を別個に扱う特徴をもっている。従って線形変数のある問題には，INTECHの方が適しているが，大体においてNS03A（HMAX＝0）と同程度の性能と考えられる。

次に射影法のPROJAルーチンだが，付録6には4次元までの計算のためNMAX＝4とした，例題2に対するテスト・プログラムが示されているが，反復計算回数（サイクル数）の上限は100，$|\underline{f}(\underline{x})|$に要求する精度ACCは$10^{-3}$と，準Newton法ルーチンと同様にして計算した。PROJAも，NS03AのHMAX＝0，INTECHと同様，ヤコビアンの微分形を与えなければならないが，計算はこれらと異なり，すべて単精度でなされている。

Table 3に例題1から8までに対する結果をまとめたが，計算は射影の方法に依存するので，考えられるすべての射影を使っている。まず1次元射影には，各反復毎に，第1次元を射影してx_1を修正し，次いで第2次元を射影してx_2を修正する(1)(2)の射影法と，これの逆の(2)(1)の射影法がある。2次元射影には，各反復毎に2つの次元を射影して(x_1, x_2)を修正する（1, 2）射影法しかないが，1次元と2次元射影の組み合わせは，Table 3に示したように(1)（1, 2），(2)（1, 2），（1, 2）(1)，（1, 2）(2)の4通りがある。

まず例題1については，1aも1bも，すべての射影の方法に対して収束している。これらの収束に要した反復計算回数をTable 4にまとめたが，1aの1次元射影のみを用いた(1)(2)，(2)(1)以外はすべて数回で正解に到達している。但し(2)（1, 2）の場合は，1aも1bも同一解に収束しているが，これらについては計算時間もNS03A（HMAX＝0），INTECHより短く問題はない。しかし1aの(1)(2)，(2)(1)の際には，3反復目で(5, −4)くらいになってから後の収束が非常に遅く，正解に収束はしているが，他の射影法の10倍程度の反復回数，計算時間を要している。次の例題2も，Table 3に示したように2aに対する(1, 2)(1)以外は収束しているか，あるいは収束値に近付いている。2aに対して(1, 2)，(2)(1, 2)，(1, 2)(2)は(−1, 2)という解に収束しているが，これもTable 1には書かれていないが正解である。2bに対する(1)(2)，(2)(1)は$|\underline{f}(\underline{x})|$がまだそれぞれ$1.2\times10^{-3}$，$3.2\times10^{-3}$で$10^{-3}$以下になっていないが，これらは更に反復を続けると，(1)(2)は103反復で(−0.70451, 1.49708)に，(2)(1)は402 msec，124反復目で(−0.70957, 1.50294)に収束する。2aの(1, 2)(1)は，3反復目で(8.30, 12.07)付近にきた後，方程式が特異となって計算が打ち切られてしまっている。計算時間は，Table 3から分るように，1次元射影のみの場合には，2次元射影をとり入れた場合の2倍から20倍も要している。2次元射影をとり入れた場合の時間は，INTECH，あるいはNS03A（HMAX＝0）と同程度，あるいはそれ以下で，PROJAの有用性を示している。

例題3については，(1, 2)，(2)(1, 2)，(1, 2)(1)，(1, 2)(2)の4つが最初から特異になって結

Table 3 Computation times (CPU msec) and solutions (in parentheses) for various projections of PROJA subroutine

Example	(1)(2)	(2)(1)	(1, 2)	(1)(1,2)
No.1a	173(3.3388, -2.9845)	172(3.3388, -2.9845)	22(3.3386, -2.9844)	20(3.3386, -2.9844)
No.1b	14(-1.53359, 0.06111)	19(-1.53344, 0.06112)	18(-1.53344, 0.06112)	15(-1.53340, 0.06113)
No.2a	39(-0.00148, 1.00067)	48(0.00144, 0.99908)	12(-1.00000, 2.00000)	18(0.00079, 0.99950)
No.2b	340(-0.70409, 1.49660)*	329(-0.71516, 1.50965)*	13(-0.70711, 1.50000)	18(-0.70711, 1.50000)
No.3	280(0.30151, 2.84185)	339(0.30374, 2.84769)*	**	14(0.29944, 2.83692)
No.4	352(-0.07016, -3.42839)*	317(-0.05715, 2.58211)*	**	**
No.5	341(0.00003, 3.78156)*	328(0.00003, 3.84599)*	**	**
No.6	315(0.46105, 0.21257)*	320(-0.64348, 0.42655)*	**	**
No.7	43(4.0000, 4.9999)	46(4.0000, 4.9995)	**	**
No.8	18(1.41421, 1.41421)	16(1.41421, 1.41421)	17(1.41421, 1.41421)	19(1.41421, 1.41421)

* Not converged even with 100 iterations
** Singular set of equations is generated

Table 3 (Continued)

Example	(2)(1,2)	(1,2)(1)	(1,2)(2)
No.1a	24(-1.53344, 0.06112)	16(3.3386, -2.9844)	21(3.3386, -2.9844)
No.1b	17(-1.53344, 0.06112)	14(-1.53344, 0.06112)	18(-1.53355, 0.06111)
No.2a	10(-1.00000, 2.00000)	**	10(-1.00000, 2.00000)
No.2b	15(-0.70711, 1.50000)	10(-0.70636, 1.49911)	11(-0.70711, 1.50000)
No.3	**	**	**
No.4	**	**	**
No.5	**	**	**
No.6	**	**	**
No.7	**	**	**
No.8	15(1.41422, 1.41421)	18(1.41421, 1.41421)	13(1.41422, 1.41421)

** Singular set of equations is generated.

果が求められていないが，(1)(1,2)ではINTECH，NS03A(HMAX=0)の半分の時間で収束している。しかし1次元射影のみの場合には収束が遅く，(2)(1)では100回反復でも $|\underline{f}(\underline{x})|$ が2.1 $\times 10^{-3}$で，115反復，395msecでようやく(0.30141，2.84187)に収束している。例題4は，2次元射影をとり入れたすべての場合，最初の反復で x_1 が0，あるいはそれに近くなり，その後で方程式が特異になって結果が求められていない。1次元射影のみの場合も $|\underline{f}(\underline{x})|$ が100回反復後で，(1)(2)の際0.070，(2)(1)の場合0.057で収束に達していない。更に200回まで反復しても両者とも殆んど動かず，(1)(2)では(−0.07015，−3.42793)，(2)(1)では(−0.05713，2.58110)になるに過ぎない。次の例題5から7までは，2次元射影法をとり入れたすべての場合，最初から方程式が特異になり不成功に終っている。また1次元射影法のみを用いた際でも，例題5，6は200回反復でも収束に到っていない。例題5の方は，Table 3に示した100回反復では $|\underline{f}(\underline{x})|$ が(1)(2)の際0.023，(2)(1)の際0.021となるが，200回反復でも，これらがそれぞれ0.014，0.013になるに過ぎず，解は(0.00002，4.27800)，(0.00002，4.30921)となるだけで殆んど改善されていない。例題6の方は更に悪く，100回反復では $|\underline{f}(\underline{x})|$ が，(1)(2)の際0.539，(2)(1)の際1.648で，特に後者の場合，収束が非常に遅い。これを200回反復までもっていっても，それぞれ0.276，0.345になり，解は(0.72391，0.52405)，(0.65600，0.42700)となるに過ぎない。例題7については，1次元射影のみの場合は，Table 4に示したように13回の反復で，INTECH，NS03A(HMAX=0)と同程度の時間で正解に収束している。

これに対し例題8は，Table 3から分るように，すべての場合に3乃至4回の反復(Table 4参照)で正解に収束しているが，計算時間は，INTECHとNS03A(HMAX=0)との中間となっている。最後の4次元の例題9については，Table 5の下に示した射影法のうち，(1)(2,3,4)，(2)(1,3,4)は最初の反復後，その他は最初から方程式が特異になり計算ができていない。1次元射影のみを用いた(1)(2)(3)(4)は，Table 5に示してあるように収束が非常に遅く，3次元と1次元の組み合わせ(1,2,3)(4)も，(1)(2)(3)(4)よりは良いが，100回反復を200回反復にしても殆んど改善がみえず，これ以上収束しそうもない。ただ(4)(1,2,3)のみが，たった2回の反復で，INTECHの殆んど1/10の時間で，大体正解を与えている。

以上から分るように，射影法のPROJAは，1次元射影のみを用いた方が解法としては安定しているが，一般に収束が非常に遅い。多次元射影をとり入れた際には，簡単な問題に対しては準Newton法のINTECH，NS03A(HMAX=0)より短時間で解が求められるが，一寸むつかしい問題では，すぐ方程式が特異となって計算ができない。例題9では，INTECHの1/10の時間で収束している例もあり，このようにうまくいく場合もあるが，まだ一般的には使えないようである。

5. 結　　論

連立非線形方程式の1つの解を，推定値から出発して反復により求める数値解法アルゴリズムを概観し，実変数の実関数に対する最近の計算プログラムの整備とベンチマーク・テストを実施した。対象としたプログラムは準Newton法のNS01A，NS03A，NONLIN，INTECH及び射

果が求められていないが，(1)(1,2)ではINTECH，NS03A(HMAX=0)の半分の時間で収束している。しかし1次元射影のみの場合には収束が遅く，(2)(1)では100回反復でも $|\underline{f}(\underline{x})|$ が2.1×10^{-3}で，115反復，395msecでようやく(0.30141，2.84187)に収束している。例題4は，2次元射影をとり入れたすべての場合，最初の反復でx_1が0，あるいはそれに近くなり，その後で方程式が特異になって結果が求められていない。1次元射影のみの場合も $|\underline{f}(\underline{x})|$ が100回反復後で，(1)(2)の際0.070，(2)(1)の場合0.057で収束に達していない。更に200回まで反復しても両者とも殆んど動かず，(1)(2)では(−0.07015，−3.42793)，(2)(1)では(−0.05713，2.58110)になるに過ぎない。次の例題5から7までは，2次元射影法をとり入れたすべての場合，最初から方程式が特異になり不成功に終っている。また1次元射影法のみを用いた際でも，例題5，6は200回反復でも収束に到っていない。例題5の方は，Table 3に示した100回反復では $|\underline{f}(\underline{x})|$ が(1)(2)の際0.023，(2)(1)の際0.021となるが，200回反復でも，これらがそれぞれ0.014，0.013になるに過ぎず，解は(0.00002，4.27800)，(0.00002，4.30921)となるだけで殆んど改善されていない。例題6の方は更に悪く，100回反復では $|\underline{f}(\underline{x})|$ が，(1)(2)の際0.539，(2)(1)の際1.648で，特に後者の場合，収束が非常に遅い。これを200回反復までもっていっても，それぞれ0.276，0.345になり，解は(0.72391，0.52405)，(0.65600，0.42700)となるに過ぎない。例題7については，1次元射影のみの場合は，Table 4に示したように13回の反復で，INTECH，NS03A(HMAX=0)と同程度の時間で正解に収束している。

これに対し例題8は，Table 3から分るように，すべての場合に3乃至4回の反復(Table 4参照)で正解に収束しているが，計算時間は，INTECHとNS03A(HMAX=0)との中間となっている。最後の4次元の例題9については，Table 5の下に示した射影法のうち，(1)(2,3,4)，(2)(1,3,4)は最初の反復後，その他は最初から方程式が特異になり計算ができていない。1次元射影のみを用いた(1)(2)(3)(4)は，Table 5に示してあるように収束が非常に遅く，3次元と1次元の組み合わせ(1,2,3)(4)も，(1)(2)(3)(4)よりは良いが，100回反復を200回反復にしても殆んど改善がみえず，これ以上収束しそうもない。ただ(4)(1,2,3)のみが，たった2回の反復で，INTECHの殆んど1/10の時間で，大体正解を与えている。

以上から分るように，射影法のPROJAは，1次元射影のみを用いた方が解法としては安定しているが，一般に収束が非常に遅い。多次元射影をとり入れた際には，簡単な問題に対しては準Newton法のINTECH，NS03A(HMAX=0)より短時間で解が求められるが，一寸むつかしい問題では，すぐ方程式が特異となって計算ができない。例題9では，INTECHの1/10の時間で収束している例もあり，このようにうまくいく場合もあるが，まだ一般的には使えないようである。

5. 結　　論

連立非線形方程式の1つの解を，推定値から出発して反復により求める数値解法アルゴリズムを概観し，実変数の実関数に対する最近の計算プログラムの整備とベンチマーク・テストを実施した。対象としたプログラムは準Newton法のNS01A，NS03A，NONLIN，INTECH及び射

Table 4 Number of iterations required for convergence for various projections of PROJA subroutine

Example	(1)(2)	(2)(1)	(1,2)	(1)(1,2)	(2)(1,2)	(1,2)(1)	(1,2)(2)
No.1a	53	54	6	5	6	4	5
No.1b	3	5	5	3	4	3	4
No.2a	11	14	3	4	2	3	2
No.2b	103	124	3	4	3	2	2
No.3	82	115	*	3	*	*	*
No.4	>200	>200	*	*	*	*	*
No.5	>200	>200	*	*	*	*	*
No.6	>200	>200	*	*	*	*	*
No.7	13	13	*	*	*	*	*
No.8	4	4	4	4	3	4	3

* Singular set of equations is generated.

Table 5 Solutions of Example No.9 obtained with various projections of PROJA*

Projection	Number of iterations	CPU msec	Obtained solution	Achieved norm
(1)(2)(3)(4)	100**	470	0.88150, -0.24066, 1.61715, 0.33059	0.151
	200**	941	0.54207, -0.40712, 1.53103, 0.28757	0.111
(1,2,3)(4)	100**	466	-0.11646, 0.51313, 1.02618, 0.75662	0.004
	200**	978	-0.11625, 0.51213, 1.02593, 0.75585	0.004
(4), (1,2,3)	2	13	-0.03564, 0.18808, 0.95377, 0.48099	0.0004

* Singular set of equations is generated for projections: (1,2),(3,4); (3,4),(1,2); (1),(2,3,4); (2,3,4),(1); (2),(1,3,4); (1,3,4),(2); (3),(1,2,4); (1,2,4),(3); (1,2,3,4).

** Not converged.

影法のPROJAの5つで，NS03Aはヤコビアン行列が疎な系を扱うように，またINTECHは線形にのみ出てくる変数を別個に扱うように特徴づけられている。これらの計算ルーチンのうち，ヤコビアンの微分形を与えなくてはならないのがINTECHとPROJA だが，NS03AもHMAX＝0のオプションを用いる際には微分形が必要である。

ベンチマーク・テストは，他の文献でも用いられている8つの2次元問題，1つの4次元問題に対してなされた。結論としては，これらの5つのルーチンの中では，BrownによるNONLIN[16]が，アルゴリズムの安定度，計算時間の両面から最善であった。ただしこのアルゴリズムは，解いていく方程式の順序に依存し，線形に近いものを先に扱うように留意しなければならない。計算量がNewton法より減少するのは方程式が複雑な時のみで，計算効率の良いのは線形に近い非線形系に対してであると記述されているが，本報の例では困難とされている問題に対して，より有用であることが顕示されている。NONLINと同様，ヤコビアンの微分形を与えなくてよいNS-01AとNS03AのHMAX＝1の2つは，性能としては大体同程度で，計算時間はNONLINの数倍以上を要求している。勿論NS03Aの特質から，ヤコビアン行列が疎な問題に対しては，NS03Aの方がNS01Aよりうまくいく場合がある。

ヤコビアンの微分形を与える計算ルーチンについては，射影法のPROJAは，準Newton法のINTECH，NS03AのHMAX＝0と比較し，多次元射影をとり入れた場合，簡単な問題に対しては短時間で解が求められるが，一寸困難な問題では，すぐ方程式が特異となって計算できなくなる。INTECHとNS03AのHMAX＝0は大体同程度の性能と考えてよいが，INTECHはその特質から線形変数をもつ系には適している。

なお最近の富士通のSSLには[27]，ヤコビアンの微分形を必要としないNewton法のルーチン，単精度のNONLESと倍精度のNONLEDがあることを付記しておく。

謝　　辞

準Newton法のNS03Aルーチンは，データが特殊な方法で格納されていたので，その整備には核設計研究室・筒井恒夫氏を煩わした。また，NONLINは同研究室の伊勢武治氏により整備されたものの提供を受け，PROJAの使用については藤村統一郎氏を煩わした。ここに，これら三氏に謝意を表す。

影法のPROJAの5つで，NS03Aはヤコビアン行列が疎な系を扱うように，またINTECHは線形にのみ出てくる変数を別個に扱うように特徴づけられている。これらの計算ルーチンのうち，ヤコビアンの微分形を与えなくてはならないのがINTECHとPROJAだが，NS03AもHMAX＝0のオプションを用いる際には微分形が必要である。

ベンチマーク・テストは，他の文献でも用いられている8つの2次元問題，1つの4次元問題に対してなされた。結論としては，これらの5つのルーチンの中では，BrownによるNONLIN[16]が，アルゴリズムの安定度，計算時間の両面から最善であった。ただしこのアルゴリズムは，解いていく方程式の順序に依存し，線形に近いものを先に扱うように留意しなければならない。計算量がNewton法より減少するのは方程式が複雑な時のみで，計算効率の良いのは線形に近い非線形系に対してであると記述されているが，本報の例では困難とされている問題に対して，より有用であることが顕示されている。NONLINと同様，ヤコビアンの微分形を与えなくてよいNS-01AとNS03AのHMAX＝1の2つは，性能としては大体同程度で，計算時間はNONLINの数倍以上を要求している。勿論NS03Aの特質から，ヤコビアン行列が疎な問題に対しては，NS03Aの方がNS01Aよりうまくいく場合がある。

ヤコビアンの微分形を与える計算ルーチンについては，射影法のPROJAは，準Newton法のINTECH，NS03AのHMAX＝0と比較し，多次元射影をとり入れた場合，簡単な問題に対しては短時間で解が求められるが，一寸困難な問題では，すぐ方程式が特異となって計算できなくなる。INTECHとNS03AのHMAX＝0は大体同程度の性能と考えてよいが，INTECHはその特質から線形変数をもつ系には適している。

なお最近の富士通のSSLには[27]，ヤコビアンの微分形を必要としないNewton法のルーチン，単精度のNONLESと倍精度のNONLEDがあることを付記しておく。

謝　　辞

準Newton法のNS03Aルーチンは，データが特殊な方法で格納されていたので，その整備には核設計研究室・筒井恒夫氏を煩わした。また，NONLINは同研究室の伊勢武治氏により整備されたものの提供を受け，PROJAの使用については藤村統一郎氏を煩わした。ここに，これら三氏に謝意を表す。

参 考 文 献

1) 朝岡卓見：“高次代数方程式の数値解法プログラム（SSLの拡充とベンチマーク・テストNo.1）”, JAERI－M 7335（1977）

2) 藤村統一郎：“連立一次方程式を解くプログラムの開発と整備（SSLの拡充とベンチマーク・テストNo.4）”, JAERI－M 7553（1978）

3) Barnes J.G.P.: "An Algorithm for Solving Non-Linear Equations Based on the Secant Method", Computer J., 8, 66 (1965)

4) Broyden C.G.: "A Class of Methods for Solving Nonlinear Simultaneous Equations", Math. Comp., 19, 577 (1965)

5) Broyden C.G.: "Quasi-Newton Methods and Their Application to Function Minimisation", Math. Comp., 21, 368 (1967)

6) Broyden C.G.: "A New Method of Solving Nonlinear Simultaneous Equations", Computer J., 12, 94 (1969)

7) Shampine L.F., Gordon M.K.: "Solving Systems of Nonlinear Equations", SAND-75-0450 (1975)

8) McCue H.K.: "Programs NAES and SS: User-Oriented Programs for Solving Nonlinear Algebraic Equations and Ordinary Differential Equations", UCID-17267 (1976)

9) Powell M.J.D.: "A Fortran Subroutine for Solving Systems of Non-Linear Algebraic Equations", AERE-R 5947 (1968)

10) Haselgrove C.B.: "The Solution of Non-Linear Equations and of Differential Equations with Two-Point Boundary Conditions", Computer J., 4, 255 (1961)

11) 富士通：“FACOM 230-60 SSL（科学用サブルーチン・ライブラリ）使用方法解説書, FORTRAN編”,（1972）

12) Reid J.K.: "Fortran Subroutines for the Solution of Sparse Systems of Non-Linear Equations", AERE-R 7293 (1972)

13) Reid J.K.: "Two Fortran Subroutines for Direct Solution of Linear Equations Whose Matrix Is Sparse, Symmetric and Positive-Definite", AERE-R 7119 (1972)

14) 藤村統一郎, 他（編）：“JSSL（原研版・科学用サブルーチン・ライブラリー）マニュアル”, JAERI－M 7102（1977）

15) Brown K.M.: "A Quadratically Convergent Newton-like Method Based upon Gaussian Elimination", SIAM J. Numer. Anal., 6, 560 (1969)

16) Brown K.M.: "Computer Oriented Algorithms for Solving Systems of Simultaneous Nonlinear Algebraic Equations", in "Numerical Solution of Systems of Nonlinear Algebraic Equations" (Byrne G.D., Hall C.A., Ed.), Academic Press (1973)

17) Boggs P.T.: "The Solution of Nonlinear Systems of Equations by A-Stable Integration Techniques", SIAM J. Numer. Anal., 8, 767 (1971)

18) Robb J.H.: "Evaluation of an Integration Technique for the Solution of Nonlinear Equations", COO-1469-226 (1973)

19) Tokko M., Keller R.F.: "Optimal Three-Dimensional Projection Method for Solving Linear Algebraic Equations", IS-3039 (1972)

20) Harms D.W., Keller R.F.: "A Direct Method Based on Projections for Solving Systems of Linear Equations", IS-3396 (1974)

21) MacEachern A., Keller R.F.: "A Projection Method for Solving Non-Linear System of Equations", IS-3040 (1972)

22) Georg D.D., Keller R.F.: "A Computer Program for Solving Non-Linear Systems of Equations", IS-3174 (1973)

23) Schmitz J.A. et al.: "Interactive Computer Programs for Linear and Non-Linear Projection Methods", IS-3179 (1973)

24) Georg D.D., Keller R.F.: "A Projection Algorithm for Solving Non-linear Systems of Equations", IS-M-16 (CONF-740511-4) (1974)

25) Georg D.D, Keller R.F.: "Stepwise Development of Algorithms for the Nonlinear Projection Methods", IS-3595 (1975)

26) Nguyen L.V., Keller R.F.: "Projection-Based Methods for Solving Systems of N Nonlinear Equations in N Unknowns", IS-3546 (1975)

27) 富士通：〝FACOM FORTRAN, SSL 使用手引書〞, (1976)

Appendix 1 FORTRAN Lists of NS01A Subroutine and its Test Program

```
ISN   ST-NO                    SOURCE PROGRAM

  1           DIMENSION X(2),F(2),AJINV(2,2),W(100)
  2           X(1)=15.
  3           X(2)=-2.
  4           N=2
  5           STEP=0.01
  6           ACC=0.000001
  7           MAXFUN=100
  8           DMAX=10.
  9           CALL CLOCKM(IIJJ)
 10           WRITE (6,202) IIJJ
 11           CALL NS01A (2,X,F,AJINV,STEP,DMAX,ACC,MAXFUN,1,W)
 12           CALL CLOCKM(IIJJ)
 13           WRITE (6,202) IIJJ
 14       202 FORMAT (5X,'TIME',I10,'MSEC')
 15           STOP
 16           END
```

```
ISN   ST-NO                    SOURCE PROGRAM

  1           SUBROUTINE CALFUN (N,X,F)
  2           DIMENSION X(1),F(1)
  3           F(1)=-13.+X(2)+((-X(1)+5.)*X(1)-2.)*X(1)
  4           F(2)=-29.+X(2)+((X(1)+1.)*X(1)-14.)*X(1)
  5           RETURN
  6           END
```

```
ISN   ST-NO                    SOURCE PROGRAM

  1           SUBROUTINE NS01A (N,X,F,AJINV,DSTEP,DMAX,ACC,MAXFUN,IPRINT,W)
  2           DIMENSION X(1),F(1),AJINV(N,N),W(1)
      C       SET VARIOUS PARAMETERS
  3           MAXC=0
      C       'MAXC' COUNTS THE NUMBER OF CALLS OF CALFUN
  4           NT=N+4
  5           NTEST=NT
      C       'NT' AND 'NTEST' CAUSE AN ERROR RETURN IF F(X) DOES NOT DECREASE
  6           DTEST=FLOAT(N+N)-0.5
      C       'DTEST' IS USED TO MAINTAIN LINEAR INDEPENDENCE
  7           NX=N*N
  8           NF=NX+N
  9           NW=NF+N
 10           MW=NW+N
 11           NDC=MW+N
 12           ND=NDC+N
      C       THESE PARAMETERS SEPARATE THE WORKING SPACE ARRAY W
 13           FMIN=0.
      C       USUALLY 'FMIN' IS THE LEAST CALCULATED VALUE OF F(X),
      C       AND THE BEST X IS IN W(NX+1) TO W(NX+N)
 14           DD=0.
      C       USUALLY DD IS THE SQUARE OF THE CURRENT STEP LENGTH
 15           DSS=DSTEP*DSTEP
 16           DM=DMAX*DMAX
 17           DMM=4.*DM
 18           IS=5
      C       'IS' CONTROLS A 'GO TO 'STATEMENT FOLLOWING A CALL OF CALFUN
 19           TINC=1.
      C       'TINC' IS USED IN THE CRITERION TO INCREASE THE STEP LENGTH
      C       START A NEW PAGE FOR PRINTING
 20           IF (IPRINT) 1,1,85
 21        85 PRINT 86
 22        86 FORMAT (1H1)
      C       CALL THE SUBROUTINE CALFUN
 23         1 MAXC=MAXC+1
 24           CALL CALFUN (N,X,F)
      C       TEST FOR CONVERCENCE
 25           FSQ=0.
 26           DO 2 I=1,N
 27           FSQ=FSQ+F(I)*F(I)
 28         2 CONTINUE
 29           IF (FSQ-ACC) 3,3,4
      C       PROVICE PRINTING OF FINAL SOLUTION IF REQUESTED
 30         3 IF (IPRINT) 5,5,6
 31         6 PRINT 7,MAXC
 32         7 FORMAT (///5X,'THE FINAL SOLUTION CALCULATED BY NS01A REQUIRED',
           1I5,' CALLS OF CALFUN, AND IS')
 33           PRINT 8,(I,X(I),F(I),I=1,N)
 34         8 FORMAT (//4X,'I',7X,'X(I)',12X,'F(I)'//(I5,2E17.8))
 35           PRINT 9,FSQ
 36         9 FORMAT (/5X,'THE SUM OF SQUARES IS',E17.8)
 37         5 RETURN
      C       TEST FOR ERROR RETURN BECAUSE F(X) DOES NOT DECREASE
 38         4 GO TO (10,11,11,10,11),IS
 39        10 IF (FSQ-FMIN) 15,20,20
 40        20 IF (DD-DSS) 12,12,11
 41        12 NTEST=NTEST-1
```

```
ISN   ST-NO                SOURCE PROGRAM     ( NS01A  )

 42         IF (NTEST) 13,14,11
 43      14 PRINT 16,NT
 44      16 FORMAT (///5X,'ERROR RETURN FROM NS01A BECAUSE',I5,
           1' CALLS OF CALFUN FAILED TO IMPROVE THE RESIDUALS')
 45      17 DO 18 I=1,N
 46         X(I)=W(NX+I)
 47         F(I)=W(NF+I)
 48      18 CONTINUE
 49         FS@=FMIN
 50         GO TO 3
       C    ERROR RETURN BECAUSE A NEW JACOBIAN IS UNSUCCESSFUL
 51      13 PRINT 19
 52      19 FORMAT (///5X,'ERROR RETURN FROM NS01A BECAUSE F(X) ',
           1'FAILED TO DECREASE USING A NEW JACOBIAN')
 53         GO TO 17
 54      15 NTEST=NT
       C    TEST WHETHER THERE HAVE BEEN MAXFUN CALLS OF CALFUN
 55      11 IF (MAXFUN-MAXC) 21,21,22
 56      21 PRINT 23, MAXC
 57      23 FORMAT(///5X,'ERROR RETURN FROM NS01A BECAUSE THERE HAVE BEEN',
           1I5,' CALLS OF CALFUN')
 58         IF (FS@-FMIN) 3,17,17
       C    PROVIDE PRINTING IF REQUESTED
 59      22 IF (IPRINT) 24,24,25
 60      25 PRINT 26,MAXC
 61      26 FORMAT (///5X,'AT THE',I5,' TH CALL OF CALFUN WE HAVE')
 62         PRINT 8,(I,X(I),F(I),I=1,N)
 63         PRINT 9,FS@
 64      24 GO TO (27,28,29,87,30),IS
       C    STORE THE RESULT OF THE INITIAL CALL OF CALUFUN
 65      30 FMIN=FS@
 66         DO 31 I=1,N
 67         W(NX+I)=X(I)
 68         W(NF+I)=F(I)
 69      31 CONTINUE
       C    CALCULATE A NEW JACOBIAN APRROXIMATION
 70      32 IC=0
 71         IS=3
 72      33 IC=IC+1
 73         X(IC)=X(IC)+DSTEP
 74         GO TO 1
 75      29 K=IC
 76         DO 34 I=1,N
 77         W(K)=(F(I)-W(NF+I))/DSTEP
 78         K=K+N
 79      34 CONTINUE
 80         X(IC)=W(NX+IC)
 81         IF (IC-N) 33,35,35
       C    CALCULATE THE INVERSE OF THE JACOBIAN AND SET THE DIRECTION MATRIX
 82      35 K=0
 83         DO 36 I=1,N
 84         DO 37 J=1,N
 85         K=K+1
 86         AJINV(I,J)=W(K)
 87         W(ND+K)=0.
 88      37 CONTINUE
 89         W(NDC+K+I)=1.
 90         W(NDC+I)=1.+FLOAT(N-I)
 91      36 CONTINUE
 92         CALL MINV2S (AJINV,N,N,.000001,ILL)
 93         IF(ILL.EQ.0) GO TO 38
 94         WRITE (6,138) ILL
 95     138 FORMAT (//5X,'STOP BY MINV2S DUE TO ILL=',I5)
 96         STOP
       C    START ITERATION BY PREDICTING THE DESCENT AND NEWTON MINIMA
 97      38 DS=0.
 98         DN=0.
 99         SP=0.
100         DO 39 I=1,N
101         X(I)=0.
102         F(I)=0.
103         K=I
104         DO 40 J=1,N
105         X(I)=X(I)-W(K)*W(NF+J)
106         F(I)=F(I)-AJINV(I,J)*W(NF+J)
107         K=K+N
108      40 CONTINUE
109         DS=DS+X(I)*X(I)
110         DN=DN+F(I)*F(I)
111         SP=SP+X(I)*F(I)
112      39 CONTINUE
       C    TEST WHETHER A NEARBY STATIONARY POINT IS PREDICTED
113         IF (FMIN*FMIN-DMM*DS) 41,41,42
       C    IF SO THEN RETURN OR REVISE JACOBIAN
114      42 GO TO (43,43,44),IS
115      44 PRINT 45
116      45 FORMAT (///5X,'ERROR RETURN FROM NS01A BECAUSE A NEARBY ',
           1'STATIONARY POINT OF F(X) IS PREDICTED')
117         GO TO 17
118      43 NTEST=0
119         DO 46 I=1,N
120         X(I)=W(NX+I)
121      46 CONTINUE
122         GO TO 32
       C    TEST WHETHER TO APPLY THE FULL NEWTON CORRECTION
123      41 IS=2
124         IF (DN-DD) 47,47,48
125      47 DD=AMAX1(DN,DSS)
126         DS=0.25*DN
127         TINC=1.
128         IF (DN-DSS) 49,58,58
129      49 IS=4
130         GO TO 80
       C    CALCULATE THE LENGTH OF STEEPEST DESCENT STEP
131      48 K=0
132         DMULT=0.
133         DO 51 I=1,N
134         DW=0.
135         DO 52 J=1,N
136         K=K+1
137         DW=DW+W(K)*X(J)
138      52 CONTINUE
139         DMULT=DMULT+DW*DW
140      51 CONTINUE
```

```
ISN   ST-NO                SOURCE PROGRAM     ( NS01A  )

141           DMULT=DS/DMULT
142           DS=DS*DMULT*DMULT
      C       TEST WHETHER TO USE THE STEEPEST DESCENT DIRECTION
143           IF (DS-DD) 53,54,54
      C       TEST WHETHER THE INITIAL VALUE OF DD HAS BEEN SET
144        54 IF (DD) 55,55,56
145        55 DD=AMAX1(DS,AMIN1(DM,DS))
146           DS=DS/(DMULT*DMULT)
147           GO TO 41
      C       SET THE MULTIPLIER OF THE STEEPEST DESCENT DIRECTION
148        56 ANMULT=0.
149           DMULT=DMULT*SQRT(DD/DS)
150           GO TO 98
      C       INTERPOLATE BETWEEN THE STEEPEST DESCENT AND THE NEWTON DIRECTIONS
151        53 SP=SP*DMULT
152           ANMULT=(DD-DS)/((SP-DS)+SQRT((SP-DD)**2+(DN-DD)*(DD-DS)))
153           DMULT=DMULT*(1.-ANMULT)
      C       CALCULATE THE CHANGE IN X AND ITS ANGLE WITH THE FIRST DIRECTION
154        98 DN=0.
155           SP=0.
156           DO 57 I=1,N
157           F(I)=DMULT*X(I)+ANMULT*F(I)
158           DN=DN+F(I)*F(I)
159           SP=SP+F(I)*W(ND+I)
160        57 CONTINUE
161           DS=0.25*DN
      C       TEST WHETHER AN EXTRA STEP IS NEEDED FOR INDEPENDENCE
162           IF (W(NDC+1)-DTEST) 58,58,59
163        59 IF (SP*SP-DS) 60,58,58
      C       TAKE THE EXTRA STEP AND UPDATE THE DIRECTION MATRIX
164        50 IS=2
165        60 DO 61 I=1,N
166           X(I)=W(NX+I)+DSTEP*W(ND+I)
167           W(NDC+I)=W(NDC+I+1)+1.
168        61 CONTINUE
169           W(ND)=1.
170           DO 62 I=1,N
171           K=ND+I
172           SP=W(K)
173           DO 63 J=2,N
174           W(K)=W(K+N)
175           K=K+N
176        63 CONTINUE
177           W(K)=SP
178        62 CONTINUE
179           GO TO 1
      C       EXPRESS THE NEW DIRECTION IN TERMS OF THOSE OF THE DIRECTION
      C       MATRIX, AND UPDATE THE COUNTS IN W(NDC+1) ETC.
180        58 SP=0.
181           K=ND
182           DO 64 I=1,N
183           X(I)=DW
184           DW=0.
185           DO 65 J=1,N
186           K=K+1
187           DW=DW+F(J)*W(K)
188        65 CONTINUE
189           GO TO (66,66),IS
190        66 W(NDC+I)=W(NDC+I)+1.
191           SP=SP+DW*DW
192           IF (SP-DS) 64,64,67
193        67 IS=1
194           KK=I
195           X(1)=DW
196           GO TO 69
197        68 X(I)=DW
198        69 W(NDC+I)=W(NDC+I+1)+1.
199        64 CONTINUE
200           W(ND)=1.
      C       REORDER THE DIRECTIONS SO THAT KK IS FIRST
201           IF (KK-1) 70,70,71
202        71 KS=NDC+KK*N
203           DO 72 I=1,N
204           K=KS+I
205           SP=W(K)
206           DO 73 J=2,KK
207           W(K)=W(K-N)
208           K=K-N
209        73 CONTINUE
210           W(K)=SP
211        72 CONTINUE
      C       GENERATE THE NEW ORTHOGONAL DIRECTION MATRIX
212        70 DO 74 I=1,N
213           W(NW+I)=0.
214        74 CONTINUE
215           SP=X(1)*X(1)
216           K=ND
217           DO 75 I=2,N
218           DS=SQRT (SP*(SP+X(I)*X(I)))
219           DW=SP/DS
220           DS=X(I)/DS
221           SP=SP+X(I)*X(I)
222           DO 76 J=1,N
223           K=K+1
224           W(NW+J)=W(NW+J)+X(I-1)*W(K)
225           W(K)=DW*W(K+N)-DS*W(NW+J)
226        76 CONTINUE
227        75 CONTINUE
228           SP=1./SQRT(DN)
229           DO 77 I=1,N
230           K=K+1
231           W(K)=SP*F(I)
232        77 CONTINUE
      C       CALCULATE THE NEXT VECTOR X, AND PREDICT THE RIGHT HAND SIDES
233        80 FNP=0.
234           K=0
235           DO 78 I=1,N
236           X(I)=W(NX+I)+F(I)
237           W(NW+I)=W(NF+I)
238           DO 79 J=1,N
239           K=K+1
240           W(NW+I)=W(NW+I)+W(K)*F(J)
241        79 CONTINUE
242           FNP=FNP+W(NW+I)**2
```

```
ISN   ST-NO                SOURCE PROGRAM      ( NSD1A  )

243       78 CONTINUE
      C      CALL CALFUN USING THE NEW VECTOR OF VERIABLES
244          GO TO 1
      C      UPDATE THE STEP SIZE
245       27 DMULT=0.9*FMIN+0.1*FNP-FS@
246          IF (DMULT) 82,81,81
247       82 DD=AMAX1(DSS,0.25*DD)
248          TINC=1.
249          IF (FS@-FMIN) 83,28,28
      C      TRY THE TEST TO DECIDE WHETHER TO INCREASE THE STEP LENGTH
250       81 SP=0.
251          SS=0.
252          DO 84 I=1,N
253          SP=SP+ABS(F(I)*(F(I)-W(NW+I)))
254          SS=SS+(F(I)-W(NW+I))**2
255       84 CONTINUE
256          PJ=1.+DMULT/(SP+ S@RT(SP*SP+DMULT*SS))
257          SP=AMIN1(4.,TINC,PJ)
258          TINC=PJ/SP
259          DD=AMIN1(DM,SP*DD)
260          GO TO 83
      C      IF F(X) IMPROVES STORE THE NEW VALUE OF X
261       87 IF(FS@-FMIN) 83,50,50
262       83 FMIN=FS@
263          DO 88 I=1,N
264          SP=X(I)
265          X(I)=W(NX+I)
266          W(NX+I)=SP
267          SP=F(I)
268          F(I)=W(NF+I)
269          W(NF+I)=SP
270          W(NW+I)=-W(NW+I)
271       88 CONTINUE
272          IF(IS-1) 28,28,50
      C      CALCULATE THE CHANGES IN F AND IN X
273       28 DO 89 I=1,N
274          X(I)=X(I)-W(NX+I)
275          F(I)=F(I)-W(NF+I)
276       89 CONTINUE
      C      UPDATE THE APPROXIMATIONS TO J AND TO AJINV
277          K=0
278          DO 90 I=1,N
279          W(MW+I)=X(I)
280          W(NW+I)=F(I)
281          DO 91 J=1,N
282          W(MW+I)=W(MW+I)-AJINV(I,J)*F(J)
283          K=K+1
284          W(NW+I)=W(NW+I)-W(K)*X(J)
285       91 CONTINUE
286       90 CONTINUE
287          SP=0.
288          SS=0.
289          DO 92 I=1,N
290          DS=0.
291          DO 93 J=1,N
292          DS=DS+AJINV(J,I)*X(J)
293       93 CONTINUE
294          SP=SP+DS*F(I)
295          SS=SS+X(I)*X(I)
296          F(I)=DS
297       92 CONTINUE
298          DMULT=1.
299          IF (ABS(SP)-0.1*SS) 94,95,95
300       94 DMULT=0.8
301       95 PJ=DMULT/SS
302          PA=DMULT/(DMULT*SP+(1.-DMULT)*SS)
303          K=0
304          DO 96 I=1,N
305          SP=PJ*W(NW+I)
306          SS=PA*W(MW+I)
307          DO 97 J=1,N
308          K=K+1
309          W(K)=W(K)+SP*X(J)
310          AJINV(I,J)=AJINV(I,J)+SS*F(J)
311       97 CONTINUE
312       96 CONTINUE
313          GO TO 38
314          END
```

Appendix 2 FORTRAN Lists of NS03A Subroutine and its Test Program for HMAX=1

```
ISN   ST-NO                SOURCE PROGRAM

  1         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  2         DIMENSION X(4),A(8),V(4),W(500),HMAX(2),IP(6),IPA(6),IRN(14),
           *      IRNA(10)
  3         M=2
  4         N=2
  5         MN=M+N
  6         X(1)=0.
  7         X(2)=1.
  8         SAC=.000001
  9         STPMIN=0.0
 10         MAXFUN=100
 11         IPRINT=-1
 12         IW=500
 13         HMAX(1)=1
 14         CALL CLOCKM(IIJJ)
 15         WRITE (6,202) IIJJ
 16         DO 20 I=1,3
 17         IP(I)=2*I-1
 18      20 IPA(I)=0
 19         DO 30 I=1,2
 20         IRN(I)=I
 21         IRN(I+2)=I
 22         IRNA(I)=0
 23      30 IRNA(I+2)=0
 24         WRITE (6,1000) (IP(I),I=1,3)
 25         WRITE (6,1000) (IRN(I),I=1,4)
 26    1000 FORMAT (5X,20I5)
 27         CALL CLOCKM(IIJJ)
 28         WRITE (6,202) IIJJ
 29          CALL      NS03A (@UNC,M,N,X,SAC,STPMIN,MAXFUN,IPRINT,W,IW,
           1 IRN,IP,A,IRNA,IPA,HMAX)
 30         CALL CLOCKM(IIJJ)
 31         WRITE (6,202) IIJJ
 32         WRITE(6,100) (X(I),I=1,2)
 33         DO 10 I=1,2
 34         CALL NS03F(I,V)
 35      10 WRITE(6,101) I,(V(J),J=1,2)
 36         CALL CLOCKM(IIJJ)
 37         WRITE (6,202) IIJJ
 38    100 FORMAT (//' FINAL X',10X,1P2E20.5)
 39    101 FORMAT (//' VARIANCE MATRIX OF COL.',I3,5X,1P2E20.5)
 40    202 FORMAT (5X,'TIME',I10,'MSEC')
 41         STOP
 42         END


ISN   ST-NO                SOURCE PROGRAM

  1         SUBROUTINE FUNC (N,X,F,M,D)
  2         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3         DIMENSION X(N),F(M),D(1)
  4         F(1)=10000.*X(1)*X(2)-1.
  5         F(2)=EXP(-X(1))+EXP(-X(2))-1.0001
  6         RETURN
  7         END


ISN   ST-NO                SOURCE PROGRAM

  1         SUBROUTINE NS03A (@UNC,M,N,X,SAC,STPMIN,MAXFUN,IPRINT,W,IW,
           1 IRN,IP,A,IRNA,IPA,HMAX)
  2          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3          REAL  STEP(2)
     C       REAL*4 X(1),W(1),A(1),LAMDA
  4          DOUBLE PRECISION
           1       X(1),W(1),A(1),LAMDA,HMAX(1)
  5          EQUIVALENCE (STEP(1),HH,IG1)
  6          INTEGER IRN(1),IP(1),IRNA(1),IPA(1)
     C       DATA ZERO/0E0/,TWO/2E0/,TEN/1E1/,EPS/1E-6 /,LENRL/4/,BIG/1E50/
     C      1,ONE/1E0/,P1/.1E0/,P9/.9E0/,P5/.5E0/
  7          DATA ZERO/0D0/,TWO/2D0/,TEN/1D1/,EPS/1D-14/,LENRL/4/,BIG/1D50/
           1,ONE/1D0/,P1/.1D0/,P9/.9D0/,P5/.5D0/
     C      LAMINC IS .TRUE. IF A SPECIAL STEP IS BEING MADE PRIOR TO A
     C  DECREASE IN D, CHANGE IS .TRUE. IF X HAS CHANGED SINCE THE LAST CALL
     C  OF TD02, UPDATE IS SET .TRUE. IF S HAS INCREASED SUFFICIENTLY
     C  LITTLE FOR IT TO BE APPROPRIATE TO USE THE SCHUBERT UPDATING FORMULA
     C  ON THE APPROXIMATE JACOBIAN. REDIFF IS .TRUE. IF ANOTHER CALL OF
     C  TD02 IS REQUIRED.
  8          LOGICAL LAMINC,CHANGE,UPDATE,ADJUST,REDIFF
  9          COMMON/TD02D /DUMMY(5),IDUMMY,ADJUST
 10          COMMON /NS03B /RHO,SIG,HEAC,FAIM,LP
     C      SET INITIAL VALUES FOR VARIOUS VARIABLES.
 11          DAIM=BIG
 12          D=BIG
 13          ALFA=(ONE-SIG)**P5
 14          REDIFF=.FALSE.
 15          ADJUST=.FALSE.
 16          DNORM=ZERO
 17          LAMINC=.FALSE.
 18          LAMDA=ZERO
 19          IT=0
 20          NFUN=0
     C
     C      PARTITION THE STORAGE IN W, DURING THE ITERATION W(NS) ONWARDS
     C IS AVAILABLE TO THE LINEAR EQUATION SOLVER.
 21          NR=0
 22          NF=0
 23          IF(IPA(1).NE.0)NF=M+1
 24          NPHI=NF+M
 25          NB=NPHI+IP(N+1)-1
 26          ND=NB+M
 27          NDC=ND+N
 28          NNR=NB
 29          IF(IPA(1).NE.0)NNR=NDC
 30          NH=NDC+M
 31          N12=(N*4+4)/LENRL
 32          IG=NH+N12
 33          NS=IG+N12+1
 34          IF(HMAX(1).EQ.ZERO)NS=NH
 35          WRITE(6,1000)M,N,IW,NR,NF,NPHI,NB,ND,NDC,NNR,NH,N12,IG,NS
 36     1000 FORMAT (5X,20I5)
 37          CALL NS03C (M,N,IRN,IP,IRNA,IPA,W,W,IW=2,NC)
     C      THIS CALL PERFORMS PRELIMINARY ANALYSIS OF THE SPARSITY STRUCTURE
     C AND HAS THE WHOLE OF W AS WORKSPACE. FOLLOWING THIS CALL W(NC).....
     C W(IW=2) IS REGARDED AS PRIVATE WORKSPACE FOR NS03C.
 38          IF(NS.GE.NC) GO TO 740
```

```
ISN   ST-NO                SOURCE PROGRAM      ( NS03A  )

 39   110   CALL FUNC(N,X,W(NF+1),M,W(NPHI+1))
 40         NFUN=NFUN+1
 41         IF(IPA(1),EQ,0) GO TO 130
 42         DO 120 I=1,M
 43   120   W(NR+I)=W(NF+I)
 44         CALL MC09A (M,N,A,X,W(NR+1),,FALSE.,IRNA,IPA)
 45   130   CALL MC02AS(W(NR+1),W(NR+1),S,M)
 46         IF(HMAX(1),EQ,ZERO) GO TO 160
      C     SET PARAMETERS FOR FIRST TD02 CALL.
 47         IG=0
 48         W(IG+1)=HH
 49         STEP(1)=HFAC*HMAX(1)
 50         STEP(2)=HFAC*HMAX(1)
 51         DO 150 I=1,N12
 52   150   W(NH+I)=HH
 53   155   IF(S,LE,SAC) GO TO 160
 54         CALL TD02A(M,N,IRN,IP,QUNC,W(NH+1),X,T,W(NF+1),-HMAX(1),W(NPHI+1),
           1 W(IG+1),W(NB+1),W(NDC+1))
 55         NFUN=NFUN+W(NB+1)
 56         CHANGE=.FALSE.
      C
      C     COMMENCE ITERATION BY PROVIDING PRINTING.
 57   160   IT=IT+1
 58         CALL MC02AS(X,X,XNORM,N)
      C     XNORM= SQRT(ANORM)
 59         XNORM=DSQRT(XNORM)
 60         IPRNTV=1
 61         IF(IPRINT,EQ,0) GO TO 180
 62         IF(MOD(IT-1,IABS(IPRINT)),NE,0)GO TO 180
 63         IPRNTV=2
 64         WRITE(LP,165)IT,NFUN,LAMDA,S,XNORM,DNORM
 65   165   FORMAT(' AFTER',I4,' ITERATIONS AND',I4,' CALLS OF FUNC LAMDA IS'
           1,1PE12,4,' AND THE SUM',
           1 ' OF SQUARES OF RESIDUALS IS',E12,4/' THE NORMS OF THE CURRENT'
           1,' ITERATE X AND THE LAST CHANGE MADE TO IT ARE',2E12,4)
 66         IF(IPRINT,GT,0) GO TO 180
 67         IPRNTV=3
 68         WRITE(LP,170)(I,X(I),I=1,N)
 69   170   FORMAT(' THE CURRENT ITERATE X IS'/5(I8,1PE16,8))
 70         WRITE(LP,175)(I,W(NR+I),I=1,M)
 71   175   FORMAT(' THE CURRENT VECTOR OF RESIDUALS IS'/ 5(I8,1PE16,8))
 72   180   IF(S,LE,SAC) GO TO 750
 73         IF(NFUN,GE,MAXFUN) GO TO 720
 74         IF(LAMINC) GO TO 190
 75         CALL NS03D (W(NPHI+1),A,LAMDA,W(NB+1))
      C     THIS CALL FACTORISES THE CURRENT MATRIX.
 76         CALL NS03E (W(NR+1),W(NB+1),W(ND+1),IPRNTV,PRED,DD,RJD,DG)
      C     THIS CALL SOLVES CURRENT LINEAR PROBLEM.
      C     DNORM= SQRT(DD)
 77         DNORM=DSQRT(DD)
 78         IF(DNORM,LE,EPS*XNORM ,OR, DNORM,LE,STPMIN) GO TO 750
 79         GO TO 200
 80   190   DO 195 I=1,M
 81   195   W(NDC+I)=-W(NB+I)
 82         IF(IPA(1),NE,6)CALL MC09A (M,N,A,W(ND+1),W(NDC+1),,FALSE.,IRNA,
           1IPA)
 83         CALL MC02AS(W(NDC+1),W(NR+1),RJD,M)
 84         CALL MC02AS(W(NDC+1),W(NDC+1),DJJD,M)
 85         U=P5
 86         IF(DJJD,GT,ZERO)U=RJD/DJJD
      C     IF( ABS(U),GT,P9)U= SIGN(P9,U)
      C     IF( ABS(U),LT,P1)U= SIGN(P1,U)
 87         IF(DABS(U),GT,P9)U=DSIGN(P9,U)
 88         IF(DABS(U),LT,P1)U=DSIGN(P1,U)
 89         DO 196 I=1,N
 90   196   W(ND+I)=X(I)-U*W(ND+I)
 91         CALL FUNC(N,W(ND+1),W(NDC+1),M,W)
 92         FDD=ZERO
 93         DO 197 I=1,M
 94         FDD=FDD+(U*W(NB+I)+W(NF+I)-W(NDC+I))**2
 95   197   W(NNR+I)=W(NDC+I)
 96         H=BIG
      C     IF(FDD,NE,ZERO)H=ALFA* ABS(U)*(ONE-U)* SQRT(DJJD/FDD)
 97         IF(FDD,NE,ZERO)H=ALFA*DABS(U)*(ONE-U)*DSQRT(DJJD/FDD)
 98         DAIM=DNORM*H
 99         REDIFF=CHANGE .AND. DNORM,LE,DAIM*FAIM
100         GO TO 291
      C     EVALUATE FUNCTION AT NEW POINT
101   200   DO 290 I=1,N
102   290   W(ND +I)=W(ND+I)  +X(I)
103         CALL FUNC(N,W(ND+1),W(NNR+1),M,W(NPHI+1))
104   291   NFUN=NFUN+1
105         IF(IPA(1),EQ,0) GO TO 295
106         DO 292 I=1,M
107   292   W(NB+I)=W(NNR+I)
108         CALL MC09A (M,N,A,W(ND+1),W(NNR+1),,FALSE.,IRNA,IPA)
109   295   CALL MC02AS(W(NNR+1),W(NNR+1),SN,M)
      C
      C     CALCULATE NEW VALUE FOR LAMDA
110         IF(LAMINC) GO TO 297
111         RATIO=(S-SN)/PRED
112         IF(RATIO,GE,ONE)RATIO=TWO-EPS-RATIO
113         IF(RATIO,GT,SIG)GO TO 320
114         IF(REDIFF ,AND, ADJUST) GO TO 750
115         IF(REDIFF)ADJUST=.TRUE.
116         IF(RATIO,GE,RHO) GO TO 300
117         IF(DNORM,GT,D*TWO) GO TO 325
118         IF(HMAX,EQ,ZERO) GO TO 303
119         D=DNORM*P5
120         IF(DNORM,GT,DAIM*FAIM) GO TO 325
121         LAMINC=.TRUE.
122         GO TO 329
123   297   BETA=BIG
124         DEN=SN-U*SO+(U-ONE)*S
125         IF(DEN,NE,ZERO)BETA=P5*(SN-U*U*SO+(U*U-ONE)*S)/DEN
      C     D=AMAX1(P1,AMIN1(P5,BETA*H))*DNORM
126         D=DMAX1(P1,DMIN1(P5,BETA*H))*DNORM
127         LAMINC=.FALSE.
128         GO TO 325
129   300   D=DNORM
130         GO TO 329
     C303   F=AMAX1(TWO,AMIN1(TEN,TWO*(S-SN)/RJD))
131    303   F=DMAX1(TWO,DMIN1(TEN,TWO*(S-SN)/RJD))
132         D=DNORM/F
```

```
ISN   ST-NO                SOURCE PROGRAM      ( NS03A  )

133           GO TO 325
      C320    D=AMIN1(D*TWO, SQRT((ONE-SIG)/(ONE-RATIO))*DNORM)
134    320    D=DMIN1(D*TWO,DSQRT((ONE-SIG)/(ONE-RATIO))*DNORM)
135           REDIFF=.FALSE.
      C325    LAMDA=AMAX1(LAMDA+(ONE-DNORM/D)*DD/DG,ZERO)
136    325    LAMDA=DMAX1(LAMDA+(ONE-DNORM/D)*DD/DG,ZERO)
137    329    UPDATE=SN.LE.S*TWO
138           IF(SN.LE.S)GO TO 335
139           IF(HMAX.EQ.ZERO) GO TO 110
140           DO 331 I=1,N
141    331    W(ND+I)=X(I)-W(ND+I)
142           DO 332 I=1,M
143    332    W(NB+I)=W(NB+I)-W(NF+I)
144           SO=SN
145           GO TO 355
      C       STORE NEW ITERATE
146    335    DO 340 I=1,N
147           G=W(ND+I)-X(I)
148           X(I)=W(ND+I)
149    340    W(ND+I)=G
150           CHANGE=.TRUE.
151           DO 350 I=1,M
152           G=W(NF+I)-W(NB+I)
153           W(NF+I)=W(NB+I)
154           W(NR+I)=W(NNR+I)
155    350    W(NB+I)=G
156           SO=S
157           S=SN
158           IF(HMAX.EQ.ZERO) GO TO 160
159    355    IF(LAMINC) GO TO 160
160           IF(REDIFF) GO TO 155
161           IF(.NOT.UPDATE) GO TO 160
      C       CORRECT THE JACOBIAN
162           CALL MC09A (M,N,W(NPHI+1),W(ND+1),W(NB+1),.FALSE.,IRN,IP)
163           DO 360 I=1,M
164    360    W(NDC+I)=ZERO
165           DO 380 J=1,N
166           K1=IP(J)
167           K2=IP(J+1)-1
168           IF(K2.LT.K1) GO TO 380
169           DO 370 K=K1,K2
170           I=IRN(K)
171    370    W(NDC+I)=W(NDC+I)+W(ND+J)**2
172    380    CONTINUE
173           DO 400 J=1,N
174           K1=IP(J)
175           K2=IP(J+1)-1
176           IF(K2.LT.K1) GO TO 400
177           DO 390 K=K1,K2
178           I=IRN(K)
179    390    IF(W(NDC+I).GT.ZERO)W(NPHI+K)=W(NPHI+K)-W(NB+I)*W(ND+J)/W(NDC+I)
180      400 CONTINUE
181           GO TO 160
      C
      C       RETURNS
182    700    WRITE(LP,710)
183    710    FORMAT('+ERROR RETURN FROM NS03 BECAUSE')
184           GO TO 750
185    720    WRITE(LP,725)
186           NFUN=MAXFUN+1
187    725    FORMAT(//32X,' MORE THAN MAXFUN CALLS OF FUNC NEEDED')
188           GO TO 700
189    740    WRITE(LP,745)
190    745    FORMAT(//32X,' WORKSPACE W IS TOO SMALL')
191           GO TO 700
      C750    W(IW)=AMAX1(W(IW),NS+IW-NC+ONE)
192     750    W(IW)=DMAX1(W(IW),NS+IW-NC+ONE)
193           W(IW-1)=NFUN
194           ADJUST=.TRUE.
195           IF(IPRNTV.NE.1 .OR. NFUN.EQ.0 .OR. IPRINT.EQ.0)RETURN
196           WRITE(LP,165)IT,NFUN,LAMDA,S,XNORM,DNORM
197           IF(IPRINT.GT.0)RETURN
198           WRITE(LP,170)(I,X(I),I=1,N)
199           WRITE(LP,175)(I,W(NR+I),I=1,M)
200           RETURN
      C
201           ENTRY NS03F (I,X)
202           IF(NFUN.GT.0 .AND. NFUN.NE.MAXFUN+1) GO TO 770
203           WRITE(LP,760)
204    760    FORMAT(' CALL TO NS03F  DOES NOT FOLLOW A SUCCESSFUL NS03A CALL')
205           RETURN
206    770    IF(LAMDA.EQ.ZERO) GO TO 780
207           LAMDA=ZERO
208           CALL NS03D (W(NPHI+1),A,LAMDA,W(NB+1))
209    780    DO 790 J=1,N
210    790    X(J)=ZERO
211           X(I)=ONE
212           CALL NS03G (X)
213           RETURN
214           END
```

```
ISN   ST-NO                SOURCE PROGRAM

  1          SUBROUTINE TD02A (M,N,IRN,IP,FUNC,H,X,Y,F,HMAX,A,IG,W,Z)
  2          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3          COMMON/TD02D /UMIN,UAIM,UMAX,EPS,EPS1,LP,ADJUST
  4          INTEGER    IRN(1),IP(1),IG(1)
  5          REAL HMAX(1),H(N),Z(1),TRUNC,ROUND,HM
  6          DIMENSION X(N),F(M),A(1),Y(1),W(1)
  7          LOGICAL REPEAT,ADJUST
     C       DATA ZERO/0./,ONE/1./
  8          DATA ZERO/0D0/,ONE/1.D0/
     C       IG(J) POINTS TO THE FIRST ENTRY OF GROUP J
  9          NFUN=0
 10          N1=N+1
     C       IG(N1+J),J=1,2,... ARE COLUMN NUMBERS FOR GROUP 1 THEN GROUP 2 ETC
 11          IF(IG(1).NE.0)GO TO 70
 12          ICC=1
 13          DO 10 J=1,N
 14          IG(J+1)=0
 15    10    W(M+J)=ZERO
     C       W(M+J) IS ZERO IF COLUMN J HAS NOT BEEN INCLUDED IN A GROUP YET
 16          DO 60 NC=1,N1
 17          IG(NC)=ICC
 18          DO 20 I=1,M
 19    20    W(I)=ZERO
     C       W(I),I=1,M HOLDS THE BOOLEAN PATTERN OF THE UNION OF THE COLUMNS
     C       SO FAR INCLUDED IN GROUP NC
 20          DO 50 J=1,N
 21          IF(W(M+J).NE.ZERO)GO TO 50
 22          K1=IP(J)
 23          K2=IP(J+1)-1
 24          IF(K2,LT,K1)GO TO 50
 25          DO 30 K=K1,K2
 26          IF(W(IRN(K)),NE,ZERO)GO TO 50
 27    30    CONTINUE
     C       ACCEPT COLUMN
 28          DO 40 K=K1,K2
 29    40    W(IRN(K))=ONE
 30          IG(N1+ICC)=J
 31          ICC=ICC+1
 32          W(M+J)=ONE
 33    50    CONTINUE
 34          IF(ICC.EQ.IG(NC))GO TO 70
 35    60    CONTINUE
     C       PRESERVE X IN W(M+1),...,W(M+N).   CHECK ALL H(J)
 36    70    DO 82 J=1,N
 37          HM=HMAX(1)
 38          IF(HM,LT,0.)HM=HMAX(J)
     C       XJ=ABS(X(J))
     C       IF(ADJUST)H(J)=(XJ+AMAX1(EPS*XJ,AMIN1(H(J),HM),EPS1*HM))-XJ
 39          XJ=DABS(X(J))
 40          IF(ADJUST)H(J)=(XJ+DMAX1(EPS*XJ,DBLE(AMIN1(H(J),HM)),EPS1*HM))-XJ
 41    82    W(J+M)=X(J)
     C       Z(J) HOLDS MAXIMUM OF ESTIMATED RATIO OF TRUNCATION ERROR TO
     C        ROUNDOFF ERROR IN COLUMN J,J=1,2,...,N
 42    83    DO 85 J=1,N
 43    85    Z(J)=1.
     C       FIND INITIAL APPROXIMATE DERIVATIVES
 44          DO 105 NG=1,N
 45          L1=IG(NG)
 46          L2=IG(NG+1)-1
 47          IF(L2,LT,L1)GO TO 110
 48          DO 90 L=L1,L2
 49          IF(H(IG(L+N1)))90,90,93
 50    90    CONTINUE
 51          GO TO 105
 52    93    DO 95 L=L1,L2
 53          J=IG(L+N1)
 54          H(J)=ABS(H(J))
 55    95    X(J)=X(J)+H(J)
 56          CALL FUNC(N,X,W,M,Y)
 57          NFUN=NFUN+1
 58          DO 100 L=L1,L2
 59          J=IG(L+N1)
 60          X(J)=W(J+M)
 61          K1=IP(J)
 62          K2=IP(J+1)-1
 63          IF(K2,LT,K1)GO TO 100
 64          DO 98 K=K1,K2
 65          I=IRN(K)
 66    98    A(K)=(W(I)-F(I))/H(J)
 67   100    CONTINUE
 68   105    CONTINUE
     C       ESTIMATE ERRORS AND IMPROVED STEP-LENGTHS
 69   110    IF(,NOT,ADJUST)GO TO 180
 70          DO 133 NG=1,N
 71          L1=IG(NG)
 72          L2=IG(NG+1)-1
 73          IF(L2,LT,L1)GO TO 135
 74          DO 113 L=L1,L2
 75          IF(H(IG(N1+L)))113,113,117
 76   113    CONTINUE
 77          GO TO 133
 78   117    DO 120 L=L1,L2
 79          J=IG(N1+L)
 80   120    X(J)=X(J)-H(J)
 81          CALL FUNC(N,X,W,M,Y)
 82          NFUN=NFUN+1
 83          DO 130 L=L1,L2
 84          J=IG(L+N1)
 85          X(J)=W(J+M)
 86          K1=IP(J)
 87          K2=IP(J+1)-1
 88          IF(K2,LT,K1)GO TO 130
 89          DO 123 K=K1,K2
 90          I=IRN(K)
 91          DER=(F(I)-W(I))/H(J)
     C       ROUND=EPS*(0.5*( ABS(W(I))+ABS(A(K)*H(J)+F(I)))+AMAX1( ABS(A(K))
     C    1, ABS(DER))*( ABS(X(J))+H(J)))/H(J)
 92          ROUND=EPS*(0.5*(DABS(W(I))+DABS(A(K)*H(J)+F(I)))+DMAX1(DABS(A(K))
             1,DABS(DER))*(DABS(X(J))+H(J)))/H(J)
 93          A(K)=(A(K)+DER)/2
 94          TRUNC=DER-A(K)
 95          IF(ROUND.EQ.0.)ROUND=1.
 96          Z(J)=AMAX1(Z(J),ABS(TRUNC/ROUND))
 97   123    CONTINUE
```

```
ISN   ST-NO                SOURCE PROGRAM      ( TD02A  )

 98   130    CONTINUE
 99   133    CONTINUE
      C      FIND IMPROVED STEPS AND DECIDE WHETHER SWEEP IS NEEDED
100   135    REPEAT=.FALSE.
101          DO 160 J=1,N
102          IF(H(J).LT.0.)GO TO 160
103          HJL=H(J)
      C      XJ=ABS(X(J))
104          XJ=DABS(X(J))
105          HM=-HMAX(1)
106          IF(HM.LT.0)HM=HMAX(J)
107          IF(Z(J).LT.UMIN)GO TO 140
      C      H(J)=(AMAX1(H(J)* SQRT(UAIM/Z(  J)), EPS*XJ,EPS1*HM)+XJ)-XJ
108          H(J)=(DMAX1(H(J)*DSQRT(UAIM/Z(  J)), EPS*XJ,EPS1*HM)+XJ)-XJ
109          IF(Z(J).GT.UMAX)GO TO 150
110   138    H(J)=-H(J)
111          GO TO 160
      C140   H(J)=(AMIN1( SQRT(UAIM/Z(J))*H(J),HM)+XJ)-XJ
      C150   IF( ABS(HJL/H(J)-1),LE.0.001 )GO TO 138
112    140   H(J)=(DMIN1(DSQRT(UAIM/Z(J))*H(J),DBLE(HM))+XJ)-XJ
113    150   IF(DABS(HJL/H(J)-1).LE.0.01D0)GO TO 138
114          REPEAT=.TRUE.
115   160    CONTINUE
116          IF(REPEAT)GO TO 83
117          DO 170 J=1,N
118   170    H(J)= ABS(H(J))
119   180    W(1)=NFUN
120          W(2)=NC-1
121          RETURN
122          ENTRY TD02B(M,N,IRN,IP,FUNC,H,X,Y,F,W,IA)
123          K=1
124          DO 200 J=1,N
125          XJ=X(J)
126          X(J)=XJ+H(J)
127          CALL FUNC(N,X,W,M,Y)
128          X(J)=XJ
129          IP(J)=K
130          DO 200 I=1,M
131          IF(W(I),EQ,F(I))GO TO 200
132          IF(K.GT.IA)GO TO 223
133          IRN(K)=I
134          K=K+1
135   200    CONTINUE
136   220    IP(N+1)=K
137          RETURN
138   223    WRITE(LP,226)
139   226    FORMAT('0ERROR RETURN FROM TD02 BECAUSE IA IS TOO SMALL')
140          IP(1)=-1
141          RETURN
142          ENTRY TD02C(N,IRN,IP,IA,MBD)
143          K=1
144          DO 230 I=1,N
145          J1=MAXO(1,I-MBD+1)
146          J2=MINO(N,I+MBD-1)
147          IP(I)=K
148          DO 230 J=J1,J2
149          IF(K.GT.IA)GO TO 223
150          IRN(K)=J
151   230    K=K+1
152          GO TO 220
153          END
```

```
ISN   ST-NO                    SOURCE PROGRAM

  1           SUBROUTINE NS03C (M,N,IRN,IP,IRNA,IPA,W,IN,IW,NS)
  2           IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3           INTEGER    IN(1),IRN(1),IPA(1),IRNA(1),IP(1)
      C       REAL*4 W(1),PHI(1),A(1),LAMDA,Y(1),R(1),D(1),S(1),V(1)
  4           DOUBLE PRECISION
             1       W(1),PHI(1),A(1),LAMDA,Y(1),R(1),D(1),S(1),V(1)
      C       DATA LENINT/2/,LENRL/4/,EPS/1E-6 /,ZERO/0E0/,ONE/1E0/
  5           DATA LENINT/2/,LENRL/4/,EPS/1D-14/,ZERO/0D0/,ONE/1D0/
  6           COMMON /NS03B /RHO,SIG,HFAC,FAIM,LP
      C     THIS ENTRY PERFORMS PRELIMINARY WORK IN ANTICIPATION OF THE
      C SOLUTION OF A SEQUENCE OF LINEARIZED PROBLEMS. W IS A WORKSPACE OF
      C DIMENSION IW AND HAS IMPLICITLY BEEN EQUIVALENCED TO IN. FOLLOWING
      C THIS ENTRY W(NS),...,W(IW) IS REGARDED AS PRIVATE WORKSPACE FOR
      C NS03D/E/F. FAILURE (BECAUSE OF LACK OF STORAGE) IS INDICATED BY NS=-1
      C AFTER A SUCCESSFUL ENTRY W(IW+2) IS SET TO INCICATE THE LEAST
      C VALUE FOR IW.
      C
      C       FIND THE SPARSITY PATTERN OF THE JACOBIAN MATRIX USING
      C  A LINKED LIST TO HOLD COLUMN NUMBERS AND ROW LINKS. IN(NP+I),I=1,M
      C  POINT TO ROW STARTS.
  7           N1=N+1
  8           IIN=(IW-LENRL)/LENINT
  9           N2=N1*2
 10           N6=N1*6
 11           NP=IIN-M
 12           IF(NP.LE.0) GO TO 740
 13           L=-1
 14           DO 10 I=1,M
 15   10      IN(NP+I)=0
 16           DO 50 J1=1,N
 17           J=N1-J1
 18           K1=IP(J)
 19           K2=IP(J+1)-1
 20           IF(K2.LT.K1) GO TO 30
 21           DO 20 K=K1,K2
 22           I=IRN(K)
 23           L=L+2
 24           IF(L.GE.NP) GO TO 740
 25           IN(L)=J
 26           IN(L+1)=IN(NP+I)
 27   20      IN(NP+I)=L
 28   30      IF(IPA(J).EQ.0) GO TO 50
 29           K1=IPA(J)
 30           K2=IPA(J+1)-1
 31           IF(K2.LT.K1)GO TO 50
 32           DO 40 K=K1,K2
 33           I=IRNA(K)
 34           IF(IN(NP+I).EQ.0) GO TO 35
 35           IF(IN(IN(NP+I)).EQ.J) GO TO 40
 36   35      L=L+2
 37           IF(L.GE.NP) GO TO 740
 38           IN(L)=J
 39           IN(L+1)=IN(NP+I)
 40           IN(NP+I)=L
 41   40      CONTINUE
 42   50      CONTINUE
      C
      C       USE A SIMILAR LINKED LIST STRUCTURE TO HOLD THE SPARSITY PATTERN
      C OF THE UPPER-TRIANGULAR PART OF THE NORMAL MATRIX, EXCLUDING THE
      C DIAGONAL.
 43           NP=IIN-N
 44           LC=NP-1
 45           DO 55 I=1,N
 46   55      IN(NP+I)=0
 47           DO 80 KI=1,L,2
 48           KJ=KI
 49           I=IN(KI)
 50           LI=IN(NP+I)
 51           LL=NP+I-1
 52   60      KJ=IN(KJ+1)
 53           IF(KJ.EQ.0)GO TO 80
 54   65      IF(LI.EQ.0) GO TO 73
 55           IF(IN(LI)-IN(KJ))70,60,73
 56   70      LL=LI
 57           LI=IN(LI+1)
 58           GO TO 65
 59   73      IN(LL+1)=LC
 60           IN(LC)=IN(KJ)
 61           IN(LC+1)=LI
 62           LL=LC
 63           LC=LC+2
 64           IF(LC-L)740,740,60
 65   80      CONTINUE
 66           LC=LC+2
 67           JW=L+1+MAXO(M,IIN-LC+1)
      C
      C       REORDER SPARSITY STRUCTURE TO FORM REQUIRED FOR MA17
 68           KN=1
 69           DO 95 I=1,N
 70           IN(I)=KN
 71           IN(N6+KN)=I
 72           KN=KN+1
 73           L=IN(NP+I)
 74   90      IF(KN.GE.LC)GO TO 740
 75           IF(L.EQ.0)GO TO 95
 76           IN(N6+KN)=IN(L)
 77           KN=KN+1
 78           L=IN(L+1)
 79           GO TO 90
 80   95      CONTINUE
 81           IN(N1)=KN
 82           L2=LENRL/LENINT+2
 83           IA=(IIN-N6)/L2
 84           JW=MAXO(JW,N6+(KN+1)*L2)
 85           NP=IW-IA
 86           IF(IA.LE.KN)GO TO 740
      C
      C        USE MA17A TO ANALYSE NORMAL MATRIX USING DUMMY MATRIX ELEMENTS.
 87           DO 105 I=1,N
 88           K1=IN(I)
 89           K2=IN(I+1)
 90           DO 100 K=K1,K2
 91   100     W(NP+K)=ZERO
 92   105     W(NP+K1)=ONE
```

```
ISN   ST-NO                SOURCE PROGRAM      ( NS03C  )

 93           CALL MA17A (W(NP+1),IN(N6+1),IN,N,N1,IA)
 94           IF(IN(N2),EQ,-3)GO TO 740
      C
      C       MOVE INTEGER ARRAYS TO END OF WORKSPACE,
 95           KA=IN(N1)-1
 96           JW=MAX0(JW,N6+KA+L2)
 97           NS = IW-KA-((KA+N2)*LENINT)/LENRL-1
 98           IF(NS,LE,0)GO TO   740
 99           INRN=IIN=KA
100           DO 107 I=1,KA
101   107     IN(INRN+I)=IN(N6+I)
102           INP=INRN=N2
103           DO 108 I=1,N2
104   108     IN(INP+I)=IN(I)
105           W(IW+2)=(JW*LENINT)/LENRL+1
106           RETURN
107   740     NS=-1
108           RETURN
      C
109           ENTRY NS03D (PHI,A,LAMDA,Y)
      C       FORM AND FACTORIZE LEAST SQUARES MATRIX, WORKSPACE Y HAS LENGTH M
110           EP=EPS
111   810     DO 820 I=1,M
112   820     Y(I)=ZERO
113           SSC=ZERO
114           DO 930 II=1,N
115           I=IN(INP+N1+II)
116           KP1=IP(I)
117           KP2=IP(I+1)-1
118           IF(KP2,LT,KP1)GO TO 835
119           DO 830 K=KP1,KP2
120   830     Y(IRN(K))=PHI(K)
121   835     IF(IPA(1),EQ,0) GO TO 850
122           KA1=IPA(I)
123           KA2=IPA(I+1)-1
124           IF(KA2,LT,KA1)GO TO 850
125           DO 840 K=KA1,KA2
126   840     Y(IRNA(K))=Y(IRNA(K))+A(K)
127   850     J1=IN(INP+I)
128           J2=IN(INP+IN(INP+N1+II+1))-1
129           DO 890 KK=J1,J2
130           J=IN(INRN+KK)
131           PROD=ZERO
132           K1=IP(J)
133           K2=IP(J+1)-1
134           IF(K2,LT,K1)GO TO 870
135           DO 860 K=K1,K2
136   860     PROD=PROD+PHI(K)*Y(IRN(K))
137   870     IF(IPA(1),EQ,0)GO TO 890
138           K1=IPA(J)
139           K2=IPA(J+1)-1
140           IF(K2,LT,K1)GO TO 890
141           DO 880 K=K1,K2
142   880     PROD=PROD+A(K)*Y(IRNA(K))
143   890     W(NS+KK)=PROD
      C       SSC=AMAX1(SSC,W(NS+J1))
144           SSC=DMAX1(SSC,W(NS+J1))
145           IF(KP2,LT,KP1)GO TO 910
146           DO 900 K=KP1,KP2
147   900     Y(IRN(K))=ZERO
148   910     IF(IPA(1),EQ,0)GO TO 930
149           IF(KA2,LT,KA1)GO TO 930
150           DO 920 K=KA1,KA2
151   920     Y(IRNA(K))=ZERO
152   930     CONTINUE
      C       ADD MULTIPLE OF IDENTITY MATRIX
      C       SSC=AMAX1(SSC*EP ,LAMDA)
153           SSC=DMAX1(SSC*EP ,LAMDA)
154           DO 935 I=1,N
155           J=NS+IN(INP+I)
156   935     W(J)=W(J)+SSC
      C       FACTORIZE NORMAL MATRIX
157           CALL MA17C (W(NS+1),IN(INRN+1),IN(INP+1),N,N1)
158           J=-IN(INP+2*N1)
159           IF(J,NE,4 ,AND, J,NE,5)RETURN
160           EP=EP*EP
161           WRITE(LP,940)
162   940     FORMAT(' MA17 ERROR IS BEING HANDLED BY NS03')
163           GO TO 810
      C
164           ENTRY NS03E (R,V,S,IPRINT,PRED,DD,DV,DG)
      C     SOLVE SYSTEM, ASSUMING FACTORIZATION KNOWN
165           DO 945 I=1,N
166   945     V(I)=ZERO
167           CALL MC09A (M,N,PHI,R,V,,TRUE,,IRN,IP)
168           IF(IPA(1),NE,0)CALL MC09A (M,N,A,R,V,,TRUE,,IRNA,IPA)
169           DO 950 I=1,N
170   950     S(I)=-V(I)
      C       PRINT VECTOR V IF REQUESTED,
171           IF(IPRINT,EQ,1)GO TO 968
172           CALL MC02AS(S,S,SS,N)
      C       SS=SQRT(SS)
173           SS=DSQRT(SS)
174           WRITE(LP,960)SS
175   960     FORMAT(' THE NORM OF THE VECTOR V IS',1PE12,4)
176           IF(IPRINT,EQ,2)GO TO 968
177           WRITE(LP,965)(I,V(I),I=1,N)
178   965     FORMAT(' THE CURRENT VECTOR V IS'/5(I8,1PE16,8))
179   968     CALL MA17B (W(NS+1),IN(INRN+1),IN(INP+1),N,N1,S,1)
180           CALL MC02AS(S,S,DD,N)
181           CALL MC02AS(S,V,DV,N)
182           DO 969 I=1,N
183   969     V(I)=-S(I)
184           CALL MA17B (W(NS+1),IN(INRN+1),IN(INP+1),N,N1,V,1)
185           CALL MC02AS(S,V,DG,N)
186           PRED=LAMDA*DD-DV
187           RETURN
      C
188           ENTRY NS03G (V)
189           CALL MA17B (W(NS+1),IN(INRN+1),IN(INP+1),N,N1,V,1)
190           RETURN
191           END
```

```
ISN   ST-NO                SOURCE PROGRAM

  1           SUBROUTINE MA17A (A,IND,IW,N,NP,IA)
  2           IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3           INTEGER   IND(1),IW(NP,6)
  4           DIMENSION A(1),B(1)
  5           COMMON/MA17D /LP
      C       DATA ZERO/0.0/
  6           DATA ZERO/0.D0/
      C       MATRIX ELEMENTS ARE HELD IN A(K),K=1,2,....,KA.
      C       ON ENTRY IND(K),K=1,KA HOLDS THE ROW NUMBER OF THE ELEMENT HELD IN
      C A(K).  IN THE MAIN BODY OF THE SUBROUTINE IND(K),IND(K+IA) HOLD THE
      C ADDRESS OF THE NEXT ELEMENT IN THE ROW/COLUMN IF THERE IS ONE AND
      C FOR THE LAST ELEMENT IN THE ROW/COLUMN HOLD (IA+ THE ROW/COLUMN
      C NUMBER).  FINALLY IND(K) IS RESET TO THE ROW NUMBER OF THE ELEMENT
      C HELD IN A(K).
      C       ON ENTRY AND ON EXIT IW(I,1) CONTAINS THE ADDRESS OF THE FIRST
      C ELEMENT OF COLUMN I AND IW(N+1,1) CONTAINS THE ADDRESS OF THE FIRST
      C UNUSED ELEMENT IN A.  ON EXIT IW(I,2) HOLDS THE COLUMN NUMBERS IN
      C UNSUCCESSFUL ENTRY IW(N+1,2)=-(ERROR NUMBER). IN THE MAIN BODY OF THE
      C SUBROUTINE IW(I,1),IW(I,2) HOLD THE ADDRESS OF THE FIRST ELEMENT OF
      C THE I TH ROW/COLUMN.
  7           KA=IW(N+1,1)-1
      C       IW(I,3) HOLDS THE POSITION IN THE ORDERING BY NUMBER OF
      C NON-ZEROS OF THE LAST ROW/COLUMN TO HAVE LESS THAN I NON-ZERO
      C ELEMENTS OR ZERO IF NONE HAVE LESS THAN I NON-ZERO ELEMENTS.
      C ON EXIT IW(I,3) HOLDS THE POSITION OF THE I TH ROW IN THE PIVOTAL
      C PIVOTAL ORDER, AFTER A SUCCESSFUL ENTRY IW(N+1,2)=N+1 AND AFTER AN
      C ORDERING.
      C       IW(I,4) HOLDS THE NUMBER OF NON-ZEROS IN THE I TH ROW/COLUMN.
      C       IW(I,5) HOLDS THE POSITION OF THE I TH ROW/COLUMN
      C IN THE ORDERING BY NUMBER OF NON-ZEROS.
      C       IW(I,6) HOLDS ROW/COLUMN NUMBERS IN PIVOTAL ORDER FOR
      C I.LT.IP AND IN ORDER OF INCREASING NUMBERS OF NON-ZEROS OTHERWISE.
  8           N1=N+1
  9           DO 10 I=1,N1
 10           IW(I,2)=IW(I,1)
 11           IW(I, 6)=I
 12           DO 10 J=3,5
 13   10      IW(I,J)=0
      C         SET ROW AND COLUMN LINKS, FIND FIRST ELEMENTS
      C OF THE ROWS AND COUNT THE NUMBER OF NON-ZEROS IN THE ROWS.
 14           DO 30 J=1,N
      C       TEMPORARILY WE USE IW(I,5) TO HOLD THE ADDRESS OF THE LAST NON-
      C ZERO ENCOUNTERED IN THE I TH ROW.
 15           K1=IW(J,2)
 16           K2=IW(J+1,2)-1
      C       CHECK THAT ALL DIAGONAL ELEMENTS ARE PRESENT.
 17           IF(IND(K1).NE.J)GO TO 600
 18           IL=0
 19           DO 20 K=K1,K2
 20           I=IND(K)
 21           IF(I.LE.IL)GO TO 520
 22           IL=I
 23           IW(I,4)=IW(I,4)+1
 24           IF(I.NE.J)IW(J,4)=IW(J,4)+1
 25           IND(K)=I+IA
 26           IND(K+IA)=K+1
 27           KL=IW(I,5)
 28           IF(KL.LE.0)IW(I,1)=K
 29           IF(KL.GT.0)IND(KL)=K
 30   20      IW(I,5)=K
 31   30      IND(K2+IA)=IA+J
      C
      C       SET UP THOSE VECTORS IN IW ASSOCIATED WITH ORDERING BY NUMBERS
      C OF NON-ZEROS.
 32           CALL KB10AS(IW(1,4),IW(1,6),N)
 33           DO 110 I=1,N
 34           J=IW(I,6)
 35           IW(J,5)=I
 36           NZ=IW(J,4)
 37   110     IW(NZ+1,3)=I
 38           J=0
 39           DO 130 I=1,N
 40           IF(IW(I,3).EQ.0)IW(I,3)=J
 41   130     J=IW(I,3)
      C
      C       NOW PERFORM THE MAIN ELIMINATION.
 42           DO 298 IP=1,N
 43           JP=IW(IP,6)
      C       JP IS THE PIVOTAL ROW/COLUMN NUMBER.
 44           KP=IW(JP,2)
      C       KP IS THE ADDRESS OF THE PIVOT.
      C
      C       MOVE THE PIVOTAL ROW/COL TO FRONT AND REPLACE ROW LINKS BY ROW
      C  NUMBERS.
 45           K=IW(JP,1)
      C       K IS USED TO SCAN THE PIVOTAL ROW/COLUMN.
 46           LL=1
      C         LL=1,2 ACCORDING AS THE ROW/COLUMN LINKS ARE NEEDED TO SCAN THE
      C PIVOTAL ROW.
 47   140     M=K
 48           LIA=IA*(2-LL)
 49   142     M=IND(M+LIA)
 50           IF(M.LE.IA)GO TO 142
 51           L=M-IA
 52           IF(L.EQ.JP)LL=2
      C       ON THE REMOVAL OF THE ELEMENT A(L,JP) THE FOLLOWING
      C INSTRUCTIONS ARE USED TO UPDATE THE NUMBERS OF ELEMENTS IN THE
      C CORRESPONDING ROW AND COLUMN AND MAKE CONSEQUENT CHANGES TO THE
      C ORDERING BY NUMBER OF NON-ZEROS.
 53           IR=L
 54           DO 144 LM=1,2
 55           NZ=IW(IR,4)-1
 56           IW(IR,4)=NZ
 57           JPOS=IW(NZ+1,3)+1
 58           IPOS=IW(IR,5)
 59           IF(IPOS.EQ.JPOS)GO TO 143
 60           JR=IW(JPOS,6)
 61           JJ=IW(IPOS,6)
 62           IW(IPOS,6)=IW(JPOS,6)
 63           IW(JPOS,6)=JJ
 64           JJ=IW(IR,5)
 65           IW(IR,5)=IW(JR,5)
 66           IW(JR,5)=JJ
 67   143     IW(NZ+1,3)=JPOS
```

```
ISN   ST-NO                  SOURCE PROGRAM      ( MA17A  )

 68           IF(L.EQ.JP)GO TO 157
 69     144   IR=JP
        C     NOW MAKE LINKS SKIP OVER PIVOTAL ROW/COLUMN.
 70           KN=IW(L,3-LL)
 71           IF(KN.NE.K)GO TO 150
 72           IW(L,3-LL)=IND(K+LIA)
 73           GO TO 155
 74     150   KL=KN
 75           KN=IND(KL+LIA)
 76           IF(KN.NE.K)GO TO 150
 77           IND(KL+ LIA)=IND(K+ LIA)
 78     155   IND(K+ IA)=IND(K+IA-LIA)
 79     157   IND(K)=L
 80           K=IND(K+IA)
 81           IF(K.LE.IA)GO TO 140
        C
        C     NOW LOOK FOR FILL-IN.
 82           K=IW(JP,1)
 83     194   L=IND(K)
 84           IF(L.EQ.JP)GO TO 290
 85           KI=IW(JP,1)
 86           KL=IW(L,1)
 87     196   JI=IND(KI)
 88           IF(JI.EQ.JP)GO TO 280
 89     199   M=KL
 90     200   M=IND(M+IA)
 91           IF(M.LE.IA)GO TO 200
 92           JL=M-IA
 93           IF(JI-JL)210,280,270
        C
        C     CREATE A NEW NON-ZERO IN POSITION (L,JI)
 94     210   KA=KA+1
 95           IF(KA.GT.IA)GO TO 580
 96           A(KA)=ZERO
 97           IR=JI
 98           IC=L
 99           DO 250 LM=1,2
100           KM=IW(IC,LM)
101           KLAST=0
102           LIA=(LM-1)*IA
103     220   M=KM
104     225   M=IND(M+IA-LIA)
105           IF(M.LE.IA)GO TO 225
106           IF(M-IA.GT.IR)GO TO 230
107           KLAST=KM
108           KM=IND(KM+LIA)
109           IF(KM.LE.IA)GO TO 220
110     230   IND(KA+LIA)=KM
111           IF(KLAST.NE.0)IND(KLAST+LIA)=KA
112           IF(KLAST.EQ.0)IW(IC,LM)=KA
113           NZ=IW(IR,4)
114           IW(IR,4)=NZ+1
115           JPOS=IW(NZ+1,3)
116           JR=IW(JPOS,6)
117           IF(IR.EQ.JR)GO TO 240
118           IPOS=IW(IR,5)
119           JJ=IW(IPOS,6)
120           IW(IPOS,6)=IW(JPOS,6)
121           IW(JPOS,6)=JJ
122           JJ=IW(IR,5)
123           IW(IR,5)=IW(JR,5)
124           IW(JR,5)=JJ
125     240   IW(NZ+1,3)=JPOS-1
126           IR=IC
127     250   IC=JI
128           KL=KA
129           GO TO 280
130     270   KL=IND(KL)
131           IF(KL-IA)199,199,290
132     280   KI=IND(KI+IA)
133           IF(KI.LE.IA)GO TO 196
134     290   K=IND(K+IA)
135           IF(K.LE.IA)GO TO 194
136     298   CONTINUE
        C
        C     SCAN THE MATRIX SETTING UP ORDERING NUMBERS IN IND(I+IA) AND
        C POINTERS TO PIVOTS IN REORDERED MATRIX IN IW(I,4).
137           J=1
138           DO 340 I=1,N
139           IP=IW(I,6)
140           IW(IP,4)=J
141           K=IW(IP,1)
142     315   KN=IND(K+IA)
143           IND(K+IA)=J
144           J=J+1
145           K=KN
146           IF(K.LE.IA)GO TO 315
147     340   CONTINUE
148           IW(N1,4)=J
        C
        C     REORDER.
149           KA=J=1
150           DO 360 I=1,KA
151           IF(I.EQ.IND(I+IA))GO TO 360
152           A1=A(I)
153           I1=IND(I)
154           J=I
155     350   K=IND(J+IA)
156           IND(J+IA)=J
157           A2=A(K)
158           I2=IND(K)
159           A(K)=A1
160           IND(K)=I1
161           A1=A2
162           I1=I2
163           J=K
164           IF(K.NE.I)GO TO 350
165     360   CONTINUE
        C
        C     SET VECTORS IN PREPARATION FOR FACTOR AND OPERATE.
166           DO 370 I=1,N1
167           IW(I,1)=IW(I,4)
168     370   IW(I,2)=IW(I,6)
        C
```

```
ISN   ST-NO                SOURCE PROGRAM      ( MA17A  )

      C     REORDER THE COLUMN ELEMENTS INTO PIVOTAL ORDER.
169         DO 375 IP=1,N
170         IC=IW(IP,2)
171         K=IW(IC,1)
172         K1=K+1
173         K2=IW(IW(IP+1,2),1)-1
174   372   K=K+1
175         IF(K.GT.K2)GO TO 375
176         IF(IW(IND(K),5).GE.IW(IND(K-1),5))GO TO 372
177         AM=A(K)
178         A(K)=A(K-1)
179         A(K-1)=AM
180         I=IND(K)
181         IND(K)=IND(K-1)
182         IND(K-1)=I
183         IF(K.NE.K1)K=K-2
184         GO TO 372
185   375   CONTINUE
      C
      C     FACTOR ENTRY
186         ENTRY MA17C (A,IND,IW,N,NP)
187         N1=N+1
188         IF(IW(N1,2).EQ.-5 .OR. IW(N1,2).EQ.-4)IW(N1,2)=N1
189         IF(IW(N1,2).NE.N1)GO TO 640
190         IFL=0
191         DO 410 IP=1,N
192         IR=IW(IP,2)
193         KP=IW(IR,1)
194         K2=IW(IW(IP+1,2),1)-1
195         IF(A(KP))377,560,378
196   377   IFL=IR
197   378   K1=KP+1
198         IF(K1.GT.K2)GO TO 410
199         DO 400 K=K1,K2
200         AL=A(K)/A(KP)
201         IC=IND(K)
202         L=IW(IC,1)-1
203         DO 390 M=K,K2
204   380   L=L+1
205         IF(IND(M).NE.IND(L))GO TO 380
206   390   A(L)=A(L)-AL*A(M)
207   400   A(K)=AL
208   410   CONTINUE
209         IF(IFL.GT.0)GO TO 620
210         RETURN
      C
      C     OPERATE ENTRY
211         ENTRY MA17B (A,IND,IW,N,NP,B,MTYPE)
212         N1=N+1
213         IF(IW(N1,2).NE.N1 .AND. IW(N1,2).NE.-5)GO TO 640
      C
      C     FORWARD SUBSTITUTION.
214         DO 420 IP=1,N
215         IC=IW(IP,2)
216         KP=IW(IC,1)
217         K1=KP+1
218         K2=IW(IW(IP+1,2),1)-1
219         IF(MTYPE.EQ.2)GO TO 417
220         IF(K1.GT.K2)GO TO 416
221         DO 415 K=K1,K2
222         IR=IND(K)
223   415   B(IR)=B(IR)-A(K)*B(IC)
224   416   B(IC)=B(IC)/A(KP)
225         GO TO 420
226   417   IF(K1.GT.K2)GO TO 419
227         DO 418 K=K1,K2
228   418   B(IC)=B(IC)+A(K)*B(IND(K))
229   419   B(IC)=B(IC)*A(KP)
230   420   CONTINUE
      C
      C     BACKWARD SUBSTITUTION
231         DO 430 I1=1,N
232         IP=1+N-I1
233         IR=IW(IP,2)
234         K1=IW(IR,1)+1
235         K2=IW(IW(IP+1,2),1)-1
236         IF(K1.GT.K2)GO TO 430
237         IF(MTYPE.EQ.2)GO TO 427
238         DO 425 K=K1,K2
239   425   B(IR)=B(IR)-A(K)*B(IND(K))
240         GO TO 430
241   427   DO 428 K=K1,K2
242         IC=IND(K)
243   428   B(IC)=B(IC)+A(K)*B(IR)
244   430   CONTINUE
245         IF(IW(N1,2).EQ.-5)GO TO 660
246         RETURN
      C
      C     THE FOLLOWING INSTRUCTIONS IMPLEMENT THE FAILURE EXITS.
247   500   WRITE(LP,510)
248   510   FORMAT('+ERROR RETURN FROM MA17 BECAUSE')
249         RETURN
250   520   WRITE(LP,530)K
251         IW(N1,2)=-1
252         GO TO 500
253   530   FORMAT(//32X,'THE ELEMENT HELD IN A(',I5,') IS OUT OF ORDER')
254   560   WRITE(LP,570)IR
255   570   FORMAT(//32X,'ZERO PIVOT FOUND IN ROW',I5)
256         IW(N1,2)=-4
257         GO TO 500
258   580   WRITE(LP,590)IP
259   590   FORMAT(//34X,'IA IS TOO SMALL, SPACE RAN OUT WHEN ELIMINATING'
     1,' ON PIVOT',I5)
260         IW(N1,2)=-3
261         GO TO 500
262   600   WRITE(LP,610)J
263   610   FORMAT(//32X,'THE',I5,'TH DIAGONAL ELEMENT IS NOT PRESENT')
264         IW(N1,2)=-2
265   620   WRITE(LP,630)
266   630   FORMAT(//32X,'RESULTS MAY BE UNRELIABLE SINCE THERE IS A NEGATIVE
     1PIVOT')
267         WRITE(LP,635)IFL
268   635   FORMAT('+',90X,'IN ROW',I5)
269         IW(N1,2)=-5
```

```
ISN   ST-NO                 SOURCE PROGRAM      ( MA17A  )

270         GO TO 500
271   640   WRITE(LP,650)
272   650   FORMAT(//32X,'PREVIOUS ENTRY GAVE ERROR RETURN')
273         GO TO 500
274   660   WRITE(LP,630)
275         GO TO 500
276         END
```

```
ISN   ST-NO                 SOURCE PROGRAM

  1         SUBROUTINE MC02AS(A,B,S,N)
  2         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3         DIMENSION A(N),B(N)
  4         S=0.0
  5         DO 1 I=1,N
  6       1 S=S+A(I)*B(I)
  7         RETURN
  8         END
```

```
ISN   ST-NO                 SOURCE PROGRAM

  1         SUBROUTINE MC09A(M,N,A,X,AX,TRANS,IRN,IP)
  2         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3         INTEGER   IRN(1),IP(1)
  4         DIMENSION A(1),X(1),AX(1)
  5         LOGICAL TRANS
  6         K=M
  7         IF(TRANS)K=N
  8         DO 50 J=1,N
  9         K1=IP(J)
 10         K2=IP(J+1)-1
 11         IF(K1.GT.K2)GO TO 50
 12         IF(TRANS)GO TO 20
 13         DO 10 K=K1,K2
 14         I=IRN(K)
 15   10    AX(I)=AX(I)+A(K)*X(J)
 16         GO TO 50
 17   20    DO 30 K=K1,K2
 18         I=IRN(K)
 19   30    AX(J)=AX(J)+A(K)*X(I)
 20   50    CONTINUE
 21         RETURN
 22         END
```

```
ISN   ST-NO                 SOURCE PROGRAM

  1         SUBROUTINE KB10AS(ITAB,INDX,N)
  2         DIMENSION ITAB(N),INDX(N)
  3         DO 1 I=1,N
  4         DO 2 J=I,N
  5         IF(ITAB(INDX(I)).LE.ITAB(INDX(J))) GO TO 2
  6         J1=INDX(J)
  7         INDX(J)=INDX(I)
  8         INDX(I)=J1
  9       2 CONTINUE
 10       1 CONTINUE
 11         RETURN
 12         END
```

```
ISN   ST-NO                 SOURCE PROGRAM

  1         BLOCK DATA
  2         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3         COMMON/TD020 /UMIN,UAIM,UMAX,EPS,EPS1,LP,ADJUST
  4         COMMON /NS036 /RHO,SIG,HFAC,FAIM,LL
  5         COMMON/MA17D /LM
  6         LOGICAL ADJUST
    C       DATA UMIN/10./, UAIM/100./, UMAX/1000./, EPS/ 1E-6/,
  7         DATA UMIN/1.D1/, UAIM/1.D2/, UMAX/1.D3 /, EPS/1.D-14/,
    C     1 LP/6/,EPS1/ 1E-6/,ADJUST/.TRUE./
          1 LP/6/,EPS1/1D-14/,ADJUST/.TRUE./
    C       DATA LL/6/,RHO/.25  /,SIG/.75  /,HFAC/.001/,FAIM/.25  /
  8         DATA LL/6/,RHO/.25D0/,SIG/.75D0/,HFAC/1D-6/,FAIM/.25D0/
  9         DATA LM/6/
 10         END
```

Appendix 3 FORTRAN List of Test Program for HMAX=0 of NS03A Subroutine

```
ISN   ST-NO                  SOURCE PROGRAM

  1          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  2          DIMENSION X(4),A(8),V(4),W(500),HMAX(2),IP(6),IPA(6),IRN(14),
            *     IRNA(10)
  3          M=2
  4          N=2
  5          MN=M+N
  6          X(1)=0.
  7          X(2)=1.
  8          SAC=.000001
  9          STPMIN=0.0
 10          MAXFUN=100
 11          IPRINT=-1
 12          IW=500
 13          HMAX(1)=0
 14          CALL CLOCKM(IIJJ)
 15          WRITE (6,202) IIJJ
 16          DO 20 I=1,3
 17          IP(I)=2*I-1
 18       20 IPA(I)=0
 19          DO 30 I=1,2
 20          IRN(I)=I
 21          IRN(I+2)=I
 22          IRNA(I)=0
 23       30 IRNA(I+2)=0
 24          WRITE (6,1000) (IP(I),I=1,3)
 25          WRITE (6,1000) (IRN(I),I=1,4)
 26     1000 FORMAT (5X,20I5)
 27          CALL CLOCKM(IIJJ)
 28          WRITE (6,202) IIJJ
 29           CALL       NS03A (@UNC,M,N,X,SAC,STPMIN,MAXFUN,IPRINT,W,IW,
            1 IRN,IP,A,IRNA,IPA,HMAX)
 30          CALL CLOCKM(IIJJ)
 31          WRITE (6,202) IIJJ
 32          WRITE(6,100) (X(I),I=1,2)
 33          DO 10 I=1,2
 34          CALL NS03F(I,V)
 35       10 WRITE(6,101) I,(V(J),J=1,2)
 36          CALL CLOCKM(IIJJ)
 37          WRITE (6,202) IIJJ
 38      100 FORMAT (//' FINAL X',10X,1P2E20.5)
 39      101 FORMAT (//' VARIANCE MATRIX OF COL.',I3,5X,1P2E20.5)
 40      202 FORMAT (5X,'TIME',I10,'MSEC')
 41          STOP
 42          END


ISN   ST-NO                  SOURCE PROGRAM

  1          SUBROUTINE FUNC (N,X,F,M,D)
  2          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
  3          DIMENSION X(N),F(M),D(1)
  4          F(1)=10000.*X(1)*X(2)-1.
  5          F(2)=EXP(-X(1))+EXP(-X(2))-1.0001
  6          D(1)     =10000.*X(2)
  7          D(2)     =-EXP(-X(1))
  8          D(3)     =10000.*X(1)
  9          D(4)     =-EXP(-X(2))
 10          RETURN
 11          END
```

Appendix 4 FORTRAN Lists of NONLIN Subroutine and its Test Program

```
ISN   ST-NO                    SOURCE PROGRAM                                SEQUENCE

      C SOLUTION OF SIMULTANEOUS NONLINEAR EQUATIONS WITHOUT REQUIRING DERIVATIVES
      C                              BY
      C                        KENNETH M, BROWN
      C     DEPARTMENT OF COMPUTER, INFORMATION, AND CONTROL SCIENCES
      C     INSTITUTE OF TECHNOLOGY        114   MAIN   ENGINEERING
      C     UNIVERSITY OF MINNESOTA
      C     MINNEAPOLIS, MINNESOTA  55455
      C SAMPLE CALLING PROGRAM
    1       DIMENSION X(2)
    2       X(1)=.4
    3       X(2)=3.
    4       CALL CLOCKM(IIJJ)
    5       WRITE (6,202) IIJJ
    6       CALL  NONLIN(2,6,100,1,X,1.E-3)
    7       CALL CLOCKM(IIJJ)
    8       WRITE (6,202) IIJJ
    9     202 FORMAT (5X,'TIME',I10,'MSEC')
   10       STOP
      C END SAMPLE CALLING PROGRAM
      C IMPORTANT!  THE USER MUST FURNISH A SUBROUTINE NAMED AUXFCN
      C             WHICH CONTAINS THE FUNCTIONS WHOSE ZEROS ARE SOUGHT.
      C             AUXFCN SHOULD RETURN THE VALUE OF THE K-TH FUNCTION
      C             OF THE SYSTEM EVALUATED AT THE VECTOR X.
      C    NOTE:    FOR MAXIMUM EFFICIENCY ORDER YOUR FUNCTIONS IN
      C             AUXFCN SO THAT THE LINEAR FUNCTIONS COME FIRST,
      C             THEN THE FUNCTIONS BECOME PROGRESSIVELY MORE
      C             NONLINEAR WITH THE "MOST NONLINEAR" FUNCTION COMING
      C             LAST.
   11       END
```

```
ISN   ST-NO                    SOURCE PROGRAM                                SEQUENCE

      CSAMPLE AUXFCN FOLLOWS
    1       SUBROUTINE AUXFCN(X,Y,K)
    2       DIMENSION X(2)
    3       GO TO (1,2),K
    4     1 Y   =(1,-.25/3.1415926536)*(EXP(2.*X(1))-2.718281828)+2.718281828*
           * X(2)/3.1415926536-2.*2.718281828*X(1)
    5       RETURN
    6     2 Y   =.5*(SIN(X(1)*X(2))-.5*X(2)/3.1415926536-X(1))
    7       RETURN
    8       END
```

```
ISN   ST-NO                    SOURCE PROGRAM                                SEQUENCE

    1         SUBROUTINE NONLIN (N,NUMSIG,MAXIT,IPRINT,X,EPS)
      C       THIS SUBROUTINE SOLVES A SYSTEM OF N SIMULTANEOUS NONLINEAR
      C       EQUATIONS, THE METHOD USED IS AT LEAST QUADRATICALLY CONVERGENT AND
      C       REQUIRES ONLY (N**2/2 + 3*N/2) FUNCTION EVALUATIONS PER ITERATIVE
      C       STEP AS COMPARED WITH (N**2 + N) EVALUATIONS FOR NEWTON'S METHOD.
      C       THIS RESULTS IN A SAVINGS OF COMPUTATIONAL EFFORT FOR SUFFICIENTLY
      C       COMPLICATED FUNCTIONS.THE METHOD DOES NOT REQUIRE THE USER TO
      C       FURNISH ANY DERIVATIVES.
      C
      C       REFERENCES:
      C       1.  K. M. BROWN, SOLUTION OF SIMULTANEOUS NONLINEAR EQUATIONS,
      C           COMM. OF THE ACM, VOL. 10, NO. 11, NOV., 1967, PP. 728-729.
      C       2.  K. M. BROWN, A QUADRATICALLY CONVERGENT NEWTON-LIKE METHOD
      C           BASED UPON GAUSSIAN ELIMINATION, SIAM J. ON NUMERICAL
      C           ANALYSIS, VOL. 6, NO. 4, DECEMBER, 1969, PP. 560-569.
      C
      C       INPUT PARAMETERS FOLLOW.
      C       N = NUMBER OF EQUATIONS ( = NUMBER OF UNKNOWNS),
      C       NUMSIG = NUMBER OF SIGNIFICANT DIGITS DESIRED,
      C       MAXIT = MAXIMUM NUMBER OF ITERATIONS TO BE ALLOWED,
      C       IPRINT = OUTPUT OPTION, OUTPUT IF = 1; HOWEVER, FAILURE
      C                INDICATIONS ARE ALWAYS OUTPUT ; MAXIT EXCEEDED AND
      C                SINGULAR JACOBIAN.
      C       X = VECTOR OF INITIAL GUESSES.
      C       EPS = CONVERGENCE CRITERION.  ITERATION WILL BE TERMINATED IF
      C             ABS(F(I)) .LT. EPS, I=1,...,N,  WHERE F(I) DENOTES THE
      C             I-TH FUNCTION IN THE SYSTEM.
      C NOTE!!  CONVERGENCE CRITERION IS CONSIDERED TO BE MET IF EITHER
      C         THE NUMBER OF SIGNIFICANT DIGITS REQUESTED IS ACHIEVED
      C         OR THE EPS CRITERION ON THE FUNCTION VALUES IS SATISFIED.
      C         TO FORCE THE ITERATION TO BE TERMINATED BY ONE OF THE
      C         CRITERIA, SIMPLY SET THE OTHER ONE TO BE VERY STRINGENT.
      C
      C       OUTPUT PARAMETERS FOLLOW..
      C       MAXIT = NUMBER OF ITERATIONS USED,
      C       X = SOLUTION OF THE SYSTEM (OR BEST APPROXIMATION THERETO).
    2         REAL              X(30),PART(30),TEMP(30),COE(30,31),RELCON,F,
             1FACTOR,HOLD,H,FPLUS,DERMAX,TEST
    3         DIMENSION ISUB(30),LOOKUP(30,30)
      C
      C       FOR EXPOSITORY PURPOSES,COE AND LOOKUP ARE DIMENSIONED AT
      C       30 X 31  AND  30 X 30  RESPECTIVELY.  CONSIDERABLE
      C       STORAGE CAN BE SAVED AT THE EXPENSE OF MAKING THE
      C       PROGRAM MORE DIFFICULT TO READ; IN FACT THE 930 LOCATIONS
      C       FOR COE REDUCE TO 495 AND THE 900 LOCATIONS FOR LOOKUP
      C       REDUCE TO JUST 30, A LISTING OF THIS CORE-WISE MORE
      C       EFFICIENT VERSION OF THE ALGORITHM IS AVAILABLE FROM THE AUTHOR.
      C
      C DELTA WILL BE A FUNCTION OF THE MACHINE AND THE PRECISION USED:
      C
    4         DELTA=1.E-7
    5         RELCON=10.E+0**(-NUMSIG)
    6         JTEST = 1
    7         IF(IPRINT .EQ. 1) PRINT 48
    8      48 FORMAT (1H1)
    9         DO 700 M = 1, MAXIT
   10         IQUIT=0
```

```
ISN   ST-NO              SOURCE PROGRAM     ( NONLIN )                    SEQUENCE

 11          FMAX = 0.
 12          M1 = M-1
 13          IF (IPRINT .NE. 1) GO TO 9
 14          PRINT    49, M1, (X(I), I = 1,N)
 15     49   FORMAT(I5,3E18.8 / (E23.8 ,2E18. 8))
 16     9    DO 10 J = 1,N
 17     10   LOOKUP (1,J) = J
       C
       C     THE ARRAY LOOKUP PERMITS A PARTIAL PIVOTING EFFECT WITHOUT HAVING
       C     TO PHYSICALLY INTERCHANGE ROWS OR COLUMNS.
       C
 18          DO 500 K = 1,N
 19          IF (K-1) 134,134,131
 20     131  KMIN = K-1
 21          CALL BACK (KMIN,N,X,ISUB,COE,LOOKUP)
       C
       C     SET UP PARTIAL DERIVATIVES OF KTH FUNCTION..
       C
 22     134  CALL AUXFCN (X,F,K)
 23          FMAX = AMAX1 (FMAX,ABS(F))
 24          IF (ABS(F) .GE. EPS) GO TO 1345
 25          I@UIT=I@UIT+1
 26          IF(I@UIT .NE. N) GO TO 1345
 27          GO TO 725
 28     1345 FACTOR=.001E+00
 29     135  ITALLY = 0
 30          DO 200 I = K,N
 31          ITEMP = LOOKUP (K,I)
 32          HOLD = X(ITEMP)
 33          PREC = 5.E-6
       C     PREC IS A FUNCTION OF THE MACHINE SIGNIFICANCE, SIG, AND SHOULD BE
       C     COMPUTED AS  PREC=5.*10.**(-SIG+2). IN THIS INSTANCE
       C     WE WERE DEALING WITH AN 8 DIGIT MACHINE.
 34          ETA = FACTOR*ABS(HOLD)
 35          H = AMIN1 (FMAX,ETA)
 36          IF (H .LT. PREC) H=PREC
 37          X (ITEMP)=HOLD+H
 38          IF (K-1)161,161,151
 39     151  CALL BACK (KMIN,N,X,ISUB,COE,LOOKUP)
 40     161  CALL AUXFCN (X,FPLUS,K)
 41          PART (ITEMP) = (FPLUS-F)/H
 42          X (ITEMP) = HOLD
 43          IF ( ABS(PART(ITEMP)) .LT. DELTA  ) GO TO 190
 44          IF ( ABS(F/PART(ITEMP)) .LE. 1.E+15)GO TO 200
 45     190  ITALLY = ITALLY+1
 46     200  CONTINUE
 47          IF (ITALLY .LE. N-K) GO TO 202
 48          FACTOR = FACTOR*10.0E+00
 49          IF (FACTOR .GT. 11.) GO TO 775
 50          GO TO 135
 51     202  IF (K .LT. N) GO TO 203
 52          IF ( ABS(PART(ITEMP)).LT. DELTA ) GO TO 775
 53          COE (K,N+1) = 0.0E+00
 54          KMAX = ITEMP
 55          GO TO 500
       C
       C     FIND PARTIAL DERIVATIVE OF LARGEST ABSOLUTE VALUE..
       C
 56     203  KMAX = LOOKUP (K,K)
 57          DERMAX = ABS (PART(KMAX))
 58          KPLUS = K+1
 59          DO 210 I = KPLUS,N
 60          JSUB = LOOKUP (K,I)
 61          TEST = ABS (PART(JSUB))
 62          IF (TEST .LT. DERMAX) GO TO 209
 63          DERMAX = TEST
 64          LOOKUP (KPLUS,I) = KMAX
 65          KMAX = JSUB
 66          GO TO 210
 67     209  LOOKUP (KPLUS,I) = JSUB
 68     210  CONTINUE
 69          IF ( ABS(PART(KMAX)) .EQ. 0.0) GO TO 775
       C
       C     SET UP COEFFICIENTS FOR KTH ROW OF TRIANGULAR LINEAR SYSTEM USED
       C     TO BACK-SOLVE FOR THE FIRST K VALUES OF X(I)
       C
 70          ISUB (K) = KMAX
 71          COE (K,N+1) = 0.0E+00
 72          DO 220 J = KPLUS,N
 73          JSUB = LOOKUP (KPLUS,J)
 74          COE (K,JSUB) = -PART (JSUB)/PART(KMAX)
 75          COE (K,N+1) = COE (K,N+1)+PART (JSUB)*X(JSUB)
 76     220  CONTINUE
 77     500  COE (K,N+1) = (COE (K,N+1)-F)/PART(KMAX)+X(KMAX)
       C     BACK SUBSTITUTE TO OBTAIN NEXT APPROXIMATION TO X:
 78          X (KMAX) = COE (N,N+1)
 79          IF (N .EQ. 1) GO TO 610
 80          CALL BACK (N-1,N,X,ISUB,COE,LOOKUP)
 81     610  IF (M-1) 650,650,625
       C
       C     TEST FOR CONVERGENCE..
       C
 82     625  DO 630 I = 1,N
 83          IF( ABS(TEMP(I)-X(I)) .GT. ABS(X(I))*RELCON ) GO TO 649
 84     630  CONTINUE
 85          JTEST = JTEST+1
 86          IF (JTEST-3)650,725,725
 87     649  JTEST = 1
 88     650  DO 660 I = 1,N
 89     660  TEMP (I) = X(I)
 90     700  CONTINUE
 91          PRINT 1753
 92     1753 FORMAT(/' NO CONVERGENCE, MAXIMUM NUMBER OF ITERATIONS USED.')
 93          IF(IPRINT .NE. 1) GO TO 800
 94          PRINT 1763
 95     1763 FORMAT('FUNCTION VALUES AT THE LAST APPROXIMATION FOLLOW:'/)
 96          IFLAG=1
 97          GO TO 7777
 98     725  IF (IPRINT .NE. 1) GO TO 800
 99     7777 DO 750 K = 1,N
100          CALL AUXFCN (X,PART(K),K)
101     750  CONTINUE
102          IF(IFLAG .NE. 1) GO TO 8777
103          PRINT 7788,(PART(K),K=1,N)
```

```
ISN   ST-NO              SOURCE PROGRAM      ( NONLIN )                  SEQUENCE

104    7788 FORMAT(3E20.8)
105         GO TO 800
106    8777 PRINT 751
107    751  FORMAT(//' CONVERGENCE HAS BEEN ACHIEVED, THE FUNCTION VALUES')
108         PRINT 7515,(PART(K),K = 1,N)
109    7515 FORMAT(' AT THE FINAL APPROXIMATION FOLLOW!'//(3E20.8))
110         GO TO 800
111    775  PRINT  752
112    752  FORMAT(//'MODIFIED JACOBIAN IS SINGULAR,  TRY A DIFFERENT')
113         PRINT 7525
114    7525 FORMAT('INITIAL APPROXIMATION,')
115    800  MAXIT=M1 + 1
116         RETURN
117         END
```

```
ISN   ST-NO              SOURCE PROGRAM                              SEQUENCE

  1         SUBROUTINE BACK (KMIN,N,X,ISUB,COE,LOOKUP)
      C     THIS SUBROUTINE BACK-SOLVES THE FIRST KMIN ROWS OF A TRIANGULARIZED
      C     LINEAR SYSTEM FOR IMPROVED X VALUES IN TERMS PREVIOUS ONES,
  2         DIMENSION        X(30),COE(30,31)
  3         DIMENSION ISUB(30),LOOKUP(30,30)
  4         DO 200 KK = 1,KMIN
  5         KM = KMIN-KK+2
  6         KMAX = ISUB (KM-1)
  7         X (KMAX) = 0.0E+00
  8         DO 100 J = KM,N
  9         JSUB = LOOKUP (KM,J)
 10         X (KMAX) = X (KMAX)+COE(KM-1,JSUB)*X(JSUB)
 11    100  CONTINUE
 12         X(KMAX)=X(KMAX)+COE(KM-1,N+1)
 13    200  CONTINUE
 14         RETURN
 15         END
```

Appendix 5 FORTRAN Lists of INTECH Subroutine and its Test Program

```
ISN   ST-NO                    SOURCE PROGRAM

      C     EXAMPLE 1A
    1       REAL*8 Y(2),DY(2),F1(2),YL(2),SAVE(2,2),YLSV(2),DEL,
     *       YD(2),PWORK(2)
    2       DIMENSION PW(2,2)
    3       NEX=1
    4       X(1)=-2.057
    5       X(2)=-7.503
    6       N=2
    7       NL=0
    8       DEL=0.001
    9       NY=N-NL
   10       NSEND=100
   11       WRITE (6,1) NEX
   12     1 FORMAT (' EXAMPLE NO.',I5//)
   13       CALL CLOCKM(IIJJ)
   14       WRITE (6,202) IIJJ
   15       CALL INTECH(Y,YL,DY,PW,DEL,F1,YD,SAVE,YLSV,PWORK,
     *  N,NY,NL,NSEND)
   16       CALL CLOCKM(IIJJ)
   17       WRITE (6,202) IIJJ
   18   202 FORMAT (5X,'TIME',I10,'MSEC')
   19       STOP
   20       END
```

```
ISN   ST-NO                    SOURCE PROGRAM

    1         SUBROUTINE DIFFUN(DY,Y,YL,N,NY,NL)
    2       REAL*8 DY(N),Y(NY),YL(NL)
    3       DY(1)=4.*Y(1)+Y(2)-Y(1)*Y(1)+2.*Y(1)*Y(2)+3.*Y(2)*Y(2)
    4       DY(2)=1.+2.*Y(1)-3.*Y(2)+Y(1)*Y(1)+Y(1)*Y(2)-2.*Y(2)*Y(2)
    5       RETURN
    6       END
```

```
ISN   ST-NO                    SOURCE PROGRAM

    1       SUBROUTINE MATSET(PW,Y,YL,N,NY,NL)
    2       REAL*8 Y(NY),YL(NL)
    3       DIMENSION PW(N,N)
    4       PW(1,1)=1.-2.*Y(1)+2.*Y(2)
    5       PW(1,2)=1.+2.*Y(1)+6.*Y(2)
    6       PW(2,1)=2.+2.*Y(1)+Y(2)
    7       PW(2,2)=-3.+Y(1)-4.*Y(2)
    8       RETURN
    9       END
```

```
ISN   ST-NO                    SOURCE PROGRAM

    1       SUBROUTINE MINV(PW,DET,N,NY,F1,PWORK)
    2       REAL*8 F1(N),PWORK(N),DET
    3       DIMENSION PW(N,N)
    4       CALL MINV2S(PW,N,N,1,E-10,ILL)
    5       IF (ILL.EQ.0) RETURN
    6       WRITE (6,20) ILL
    7    20 FORMAT (5X,'FAILED IN MINV2S WITH ILL =',I7)
    8       STOP
    9       END
```

```
ISN   ST-NO                    SOURCE PROGRAM

    1       SUBROUTINE MATMUL(PW,DY,F1,N)
    2       REAL*8 F1(N),DY(N),SUM
    3       DIMENSION PW(N,N)
    4       DO 50 J=1,N
    5       SUM=0
    6       DO 40 K=1,N
    7    40 SUM=SUM+PW(J,K)*DY(K)
    8    50 F1(J)=SUM
    9       RETURN
   10       END
```

```
ISN   ST-NO                SOURCE PROGRAM

  1          SUBROUTINE INTECH(Y,YL,DY,PW,DEL,F1,YD,SAVE,YLSV,
             * PWORK,N,NY,NL,NSEND)
  2          REAL*8 DY(N),F1(N),Y(NY),YL(NL),YD(NY),SAVE(2,NY),YLSV(NL),
             * PWORK(N),DMAX1,DMIN1,DEL,ALPHA,H,R,DET,SS,S,HOLD,DD,D,HH
  3          DIMENSION PW(N,N)
      C
      C      PARAMETER LIST
      C         Y  VECTOR OF NONLINEAR VARIABLES (INPUT)
      C         YL VECTOR OF LINEAR VARIABLES (INPUT)
      C         DY VECTOR OF FUNCTION VALUES , CALCULATED
      C               BY THE SUBROUTINE DIFFUN
      C         PW JACOBIAN MATRIX CALCULATED BY SUBROUTINE MATSET
      C               OR THE INVERS JACOBIAN AFTER CALLING MINV
      C         DEL ERROR CRITERION , SUPPLIED BY CALLING PROGRAM
      C         F1 VECTOR EQUAL TO PRODUCT OF PW AND DY
      C         YD THE VECTOR OF CHANGES TO BE APPLIED TO VECTOR Y
      C         SAVE TWO ROW VECTOR USED TO STORE VALUES OF Y AND YD
      C               BEFORE NEXT ITERATION , IN CASE RE-START NEEDED
      C         YLSV VECTOR USED TO STORE VALUES OF YL BEFORE
      C               NEXT ITERATION , IN CASE RE-START NEEDED
      C         PWORK USED ONLY FOR STORAGE SPACE WITHIN SUBROUTINE MINV
      C         N THE TOTAL NUMBER OF VARIABLES (INPUT)
      C         NY THE NUMBER OF NONLINEAR VARIABLES (INPUT)
      C         NL THE NUMBER OF LINEAR VARIABLES (INPUT)
      C         NSEND THE MAXIMUM NUMBER OF ITERATION DESIRED (INPUT)
      C
      C      INITIALIZED VARIABLES
      C
  4          NOFAIL=1
  5          JACOB=0
  6          R=1
  7          NZ=N*N
  8          NJ=NY*2
  9          IF(NJ.LT.10) NJ=10
 10          NJJ=NJ*2/3
 11          ALPHA=0
 12          H=1
 13          NS=0
 14          NW=0
      C
      C      PRINT HEADING
      C
 15          WRITE (6,1) N,NL,DEL
 16        1 FORMAT(5X,' N =',I4,'   NL =',I4,'   DEL =',D10.2)
 17          WRITE (6,2)
 18        2 FORMAT(5X,'  NS NW  ALPHA      H',8X,'ERROR ',6X,
             * 'Y(J) AND YL(*)'//)
      C
      C      INITIAL EVALUATION OF FUNCTION (DY)
      C
 19          NS=NS+1
 20          CALL DIFFUN (DY,Y,YL,N,NY,NL)
      C
      C      INITIAL EVALUATION OF JACOBIAN (PW)
      C
 21          CALL MATSET (PW,Y,YL,N,NY,NL)
      C
      C      INITIAL EVALUATION OF INVERSE JACOBIAN (PW)
      C
 22          NW=NW+1
 23          CALL MINV(PW,DET,N,NZ,F1,PWORK)
      C
      C      PRODUCT PW*DY IN VECTOR F1
      C
 24          CALL MATMUL (PW,DY,F1,N)
      C
      C      NORM STORED IN SS FOR STARTING VALUES
      C
 25          SS=0
 26          DO 3 J=1,N
 27        3 SS=SS+DABS(DY(J))
      C
      C      PRINT RESULTS OBTAINED WITH STARTING VECTOR
      C
 28          WRITE (6,13) NS,NW,ALPHA,H,SS,(Y(J),J=1,NY)
 29          IF(NL.GT.0) WRITE(6,14) (YL(J),J=1,NL)
      C
      C      PREPARE FOR THE INITIAL PREDICTOR STEP FOR Y.  DO THE
      C            INITIAL CORRECTOR STEP FOR YL
      C
 30          DO 4 J=1,NY
 31        4 YD(J)=-H*F1(J)
 32          IF(NL.LE.0) GO TO 6
 33          DO 5 J=1,NL
 34        5 YL(J)=YL(J)-F1(J+NY)
 35        6 CONTINUE
      C
      C      SAVE PRESENT VECTORS Y AND YL
      C
 36          DO 7 J=1,NY
 37          SAVE(1,J)=Y(J)
 38        7 SAVE(2,J)=YD(J)
 39          IF(NL.LE.0) GO TO 9
 40          DO 8 J=1,NL
 41        8 YLSV(J)=YL(J)
 42        9 CONTINUE
      C
      C      INDICATE NEWTON'S METHOD IS IN USE BY NEWTON = 1
      C
 43          NEWTON=1
      C
      C      BEGIN MAIN ITERATION LOOP BY COMPLETING THE PREDICTION OF Y
      C
 44       10 DO 11 J=1,NY
 45       11 Y(J)=Y(J)+YD(J)
      C
      C      EVALUATE THE FUNCTION (DY)
      C
 46          NS=NS+1
 47          CALL DIFFUN (DY,Y,YL,N,NY,NL)
      C
      C      CALCULATE THE VALUE OF THE NORM (S)
      C
 48          S=0
```

```
ISN   ST-NO                SOURCE PROGRAM      ( INTECH )

 49          DO 12 J=1,N
 50       12 S=S+DABS(DY(J))
      C
      C      PRINT RESULTS FOLLOWING PREDICTOR STEP , THEN RETURN IF
      C         NORM IS LESS THAN OR EQUAL TO DEL
      C
 51          WRITE (6,13) NS,NW,ALPHA,H,S,(Y(J),J=1,NY)
 52       13 FORMAT (5X,I4,I3,F6.2,2D11.2,2D15.5/(10X,4D15.5))
 53          IF(NL.GT.0) WRITE (6,14) (YL(J),J=1,NL)
 54       14 FORMAT (10X,4D15.5)
 55          IF(S.LE.DEL) RETURN
      C
      C      CALCULATE PRED (RATIO OF NORM / PREVIOUS NORM)
      C
 56          PRED=S/SS
      C
      C      BEGIN PROCESS OF DETERMINING VALUES OF ALPHA AND R (FACTOR
      C         BY WHICH H WILL BE MULTIPLIED) FOR NEXT ITERATION
      C
 57          IF(NEWTON.EQ.1) GO TO 35
 58          IF(PRED.LT.100) GO TO 15
 59          IF(NOFAIL.EQ.1) GO TO 30
 60       15 CONTINUE
 61          IF(PRED.LT,.98) GO TO 16
 62          ALPHA=1
 63          R=DMIN1(1,3D0*D,6D0/H)
 64          GO TO 17
 65       16 R=1.7D0-.65D0*H+.15D0/H
 66          ALPHA=0.8*ALPHA
      C
      C      SAVE PRESENT VECTORS AND VALUE OF H BEFORE NEXT ITERATION
      C
 67       17 DO 18 J=1,NY
 68          SAVE(1,J)=Y(J)
 69       18 SAVE(2,J)=YD(J)
 70          IF(NL.LE.0) GO TO 20
 71          DO 19 J=1,NL
 72       19 YLSV(J)=YL(J)
 73       20 CONTINUE
 74          HOLD=H
      C
      C      DETERMINE WHETHER JACOBIAN SHOULD BE RE-EVALUATED
 75          IF(SS.GE.1.D0) GO TO 21
 76          IF(JACOB.LT.NJJ) GO TO 22
 77       21 CONTINUE
 78          JACOB=JACOB-1
 79          IF(JACOB.GT.0) GO TO 23
      C
      C      EVALUATE THE JACOBIAN (PW)
      C
 80       22 CALL MATSET (PW,Y,YL,N,NY,NL)
 81          NW=NW+1
      C
      C      EVALUATE THE INVERSE OF JACOBIAN (PW)
      C
 82          CALL MINV(PW,DET,N,NZ,F1,PWORK)
 83          JACOB=NJ
      C
      C      FORM VECTOR F1 , THE PRODUCT PW*DY , THEN DO THE
      C         CORRECTOR STEP FOR Y
      C
 84       23 CALL MATMUL (PW,DY,F1,N)
 85          IF(ALPHA.LT.0.01) GO TO 25
 86          DD=1.D0/(1+H*ALPHA)
 87          DO 24 J=1,NY
 88          D=(F1(J)*H+YD(J))*DD
 89          Y(J)=Y(J)-ALPHA*D
 90       24 YD(J)=R*(YD(J)-D)
 91          GO TO 27
 92       25 HH=-H*R
 93          DO 26 J=1,NY
 94       26 YD(J)=HH*F1(J)
 95       27 CONTINUE
 96          IF(NL.LE.0) GO TO 29
      C
      C      DO THE CORRECTOR STEP FOR THE LINEAR VARIABLES IN YL
      C
 97          DO 28 J=1,NL
 98       28 YL(J)=YL(J)-F1(J+NY)
 99       29 SS=S
100          H=H*R
101          NOFAIL=1
102          IF(NS.GE.NSEND) GO TO 37
      C
      C      END OF MAIN ITERATION LOOP
      C
103          GO TO 10
104       30 CONTINUE
      C
      C      CHANGE ALPHA, R, AND H FOR THE CASE (PRED.GT.100 AND NOFAIL=1)
      C
105          ALPHA=1.0
106          IF(JACOB.LT.NJJ) JACOB=0
107          R=DMAX1(0.5D0,0.2D0/HOLD)
108          H=HOLD*R
      C
      C      PREVIOUS VECTORS RESTORED AND NOFAIL SET TO ZERO BEFORE
      C         RE-STARTING LAST STEP , BECAUSE OF LARGE INCREASE IN ERROR
      C
109       31 DO 32 J=1,NY
110          Y(J)=SAVE(1,J)
111       32 YD(J)=SAVE(2,J)*R
112          IF(NL.LE.0) GO TO 34
113          DO 33 J=1,NL
114       33 YL(J)=YLSV(J)
115       34 CONTINUE
116          NOFAIL=0
117          GO TO 10
118       35 CONTINUE
      C
      C      IF (NEWTON=1 AND PRED.LT.95) GO BACK TO DO THE NORMAL
      C         CORRECTION PROCESS , REMAINING IN THE NEWTON METHOD
119          IF (PRED.LT.0.95) GO TO 17
      C
```

```
ISN   ST-NO                  SOURCE PROGRAM      ( INTECH )

      C     BUT IF PRED.GE.0.95 PRINT A MESSAGE, CHANGE ALPHA, H, R,
      C        NEWTON, JACOB, THEN GO TO RESTOR PREVIOUS VECTORS AND
      C        AFTER VALUE OF NOFAIL BEFORE RE-STARTING LAST ITERATION
      C
120           WRITE (6,36) S,(Y(J),J=1,NY)
121        36 FORMAT (5X,'NEWTON FAILED,',9X,D11.2,2D15.5/(10X,4D15.5))
122           IF(NL.GT.0) WRITE(6,14) (YL(J),J=1,NL)
123           H=.01
124           ALPHA=1
125           R=.01
126           JACOB=0
127           NEWTON=0
128           GO TO 31
129        37 CONTINUE
      C
      C     IF THE NUMBER OF FUNCTION EVALUATIONS HAS EXCEEDED NSEND,
      C        PRINT THE VECTORS DY, Y, YL, AND F1, THEN RETURN
130           WRITE (6,38) (DY(J),J=1,N)
131        38 FORMAT (5X,'-DY,'/(5X,'**',4D14.6))
132           WRITE (6,39) (Y(J),J=1,NY)
133        39 FORMAT (5X,'-Y,'/(5X,'**',4D16.8))
134           WRITE (6,40) (J,F1(J),J=1,N)
135        40 FORMAT (5X,'-F1,'/3(5X,I6,'**',D14.6))
136           RETURN
137           END
```

Appendix 6 FORTRAN List of Test Program for PROJA Subroutine

```
ISN   ST-NO                    SOURCE PROGRAM

  1           REAL*4 JFX(4,4)
  2           DIMENSION NDIM(4),NCOL(4,4),A(4,4),B(4),F(4),X(4)
  3         5 READ (5,100,END=99) NMAX
  4           CALL CLOCKM(IIJJ)
  5           WRITE (6,202) IIJJ
  6           CALL PROJA(NDIM,NCOL,A,B,F,JFX,X,NMAX)
  7           CALL CLOCKM(IIJJ)
  8           WRITE (6,202) IIJJ
  9           GO TO 5
 10       100 FORMAT (I10)
 11       202 FORMAT (5X,'TIME',I10,'MSEC')
 12        99 STOP
 13           END
```

```
ISN   ST-NO                    SOURCE PROGRAM

  1           SUBROUTINE PFUNC(X,F,N)
  2           DIMENSION X(4),F(4)
  3           F(1)=X(1)*X(1)-X(2)+1.
  4           F(2)=X(1)-COS(3.1415926536/2.*X(2))
  5           RETURN
  6           END
```

```
ISN   ST-NO                    SOURCE PROGRAM

  1           SUBROUTINE JCBN(JFX,X,N)
  2           REAL*4 JFX(4,4)
  3           DIMENSION X(4)
  4           JFX(1,1)=2.*X(1)
  5           JFX(1,2)=-1.
  6           JFX(2,1)=1.
  7           JFX(2,2)=SIN(3.1415926536/2.*X(2))*3.1415926536/2.
  8           RETURN
  9           END
```