28
6-9-75
5-NTIS
IS-3595

STEPWISE DEVELOPMENT
OF ALGORITHMS FOR THE
NONLINEAR PROJECTION METHODS

D. D. Georg and R. F. Keller

AMES LABORATORY, USERDA
IOWA STATE UNIVERSITY
AMES, IOWA

MASTER

Date Transmitted: April 1975

PREPARED FOR THE U. S. ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
UNDER CONTRACT W-7405-eng-82

DISTRIBUTION OF THIS DOCUMENT UNLIMITED

DISCLAIMER

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

DISCLAIMER

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

STEPWISE DEVELOPMENT OF ALGORITHMS FOR
THE NONLINEAR PROJECTION METHODS

By

D. D. Georg and R. F. Keller

Ames Laboratory, ERDA
Iowa State University
Ames, Iowa 50010

Date Transmitted: April 1975

NOTICE

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

PREPARED FOR THE U.S. ENERGY RESEARCH AND
DEVELOPMENT ADMINISTRATION UNDER CONTRACT NO. W-7405-eng-82

MASTER

DISTRIBUTION OF THIS DOCUMENT UNLIMITED

Page

—NOTICE—

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

Available from: National Technical Information Service
U. S. Department of Commerce
P.O. Box 1553
Springfield, VA 22161

Price: Microfiche \$2.25
 Paper Copy \$3.75

TABLE OF CONTENTS

	PAGE
ABSTRACT	iv
I. INTRODUCTION	1
II. STEPWISE DEVELOPMENT OF THE NONLINEAR PROJECTION ALGORITHM	5
III. STEPWISE DEVELOPMENT OF THE SUBSPACE SELECTION CRITERIA	13
IV. BIBLIOGRAPHY	23

ABSTRACT

The nonlinear projection methods are a class of minimization methods for solving nonlinear systems of equations. At each iteration step, from one to n , the dimension of the system, components can be modified. A particular nonlinear projection method is determined by the sequence of changes which are made in the approximations to the solution vector.

In this report, algorithms are developed using the top-down stepwise refinement approach for the projection methods and for the mechanisms which are used to dynamically determine sets of components to be modified at each iteration step. The final algorithms are in a form that can be easily translated into an arbitrary programming language.

Distribution List

D. D. Georg	13
R. F. Keller	13
OR	27
ERDA Chicago Patent Office	1
Ames Laboratory Library	<u>6</u>
Total	60

I. INTRODUCTION

The nonlinear projection method which was developed by MacEachern[2] and extended by White[3] is an iterative method for solving nonlinear systems of equations which allows the user to modify from one to n , the dimension of the system, components of the approximate solution vector at each iteration step. New approximations to the solution vector are successively obtained by solving for change vectors d^k , $k = 0, 1, \dots$, and then modifying the previous approximate solution vectors x^k , $k = 0, 1, \dots$.

$$x^{k+1} = x^k + d^k \quad k = 0, 1, \dots$$

If $B_k = \{i_1, i_2, \dots, i_{n_k}\}$ is the set of n_k components that are being modified at step k , then in order to obtain d^k , it is first necessary to solve the following linear system to obtain dx^k .

$$\sum_{m \in B_k} (J_m^k, J_r^k) dx_m^k = - (F(x^k), J_r^k)$$

$$r \in B_k$$

Note that J_m^k is the m th column of the Jacobian matrix associated with the system F evaluated at x^k . After solving the system, the nonzero components of d^k are obtained directly from dx^k .

The sequence $\{B_k\}$ that is used to solve a nonlinear system $F(x) = 0$ does affect the rate of convergence of the projection method. In fact, it is easy to demonstrate experimentally that the amount of reduction obtained in the Euclidean norm of the residue vector $F(x^k)$

at each iteration step k is dependent on the particular B_k which is used.

How is B_k determined at each step k ? In the work recently completed by Georg[1], criteria are developed for determining quasi-optimal B_k . The criteria developed by Georg are intended to be used dynamically. That is, at each iteration step, calculations are performed to determine which components should be modified.

Following the modification of the n_k components in B_k at step k , the residue vector associated with the new approximate solution vector, $F(x^{k+1})$, is perpendicular to n_k columns of the Jacobian matrix evaluated at the previous approximate solution vector x^k . These n_k columns correspond exactly to the components which were modified at step k . This fact not only suggests the name, "projection method", but it also motivates two of the criteria to be presented in this paper.

The angle criteria chooses projection subspaces, i.e., sets of columns to be projected on or sets of components to be modified, on the basis of the cosines of the angles between the columns of the Jacobian matrix and the residue vector. A particular column i is included in the projection subspace, B_k , at step k if

$$\gamma_i = \cos^2 \theta_{r J_i^k} > T$$

where $\theta_{r J_i^k}$ is the angle between $r^k = F(x^k)$ and the i th column of the Jacobian matrix evaluated at x^k and T is a pre-determined tolerance.

For example, if $T = .25$, then no column would be projected on if the residue vector was within 30° of being perpendicular to it.

The angle criteria as well as the other criteria that will be discussed could potentially choose a B_k that is empty. In this case, the projection algorithms that use dynamic subspace selection criteria must specify what action is to be taken. It will be assumed in this paper, in the event that a criteria chooses B_k to be empty, that all the components of the approximate solution vector will be modified at step k .

Another criteria, which is known as the φ criteria, also attempts to take advantage of the geometrical relationship between $F(x^{k+1})$ and the columns of the Jacobian matrix evaluated at x^k . This criteria includes column i in B_{k+1} if

$$(\Delta_i^k, \Delta_i^k) > (J_i^k, J_i^k)V$$

where $\Delta_i^k = J_i^{k+1} - J_i^k$ and V is a pre-determined tolerance. Because of the use of the Δ_i^k , the φ criteria must use some other basis to determine its initial subspace, B_0 .

The tolerances, T for the angle criteria and V for the φ criteria, do influence the effectiveness of the respective criteria to choose optimal projection subspaces. An algorithm that chooses subspaces using the angle criteria or φ criteria must either select a single value to be used for the tolerance at each iteration step or else dynamically determine a different tolerance value at each step. In

this paper, projection algorithms that use the angle criteria or φ criteria will select a single value for the tolerance.

The final criteria to be discussed in this paper is the τ criteria. This criteria was developed as a result of an attempt to maximize an approximation for $\Delta r^k = ||r^k|| - ||r^{k+1}||$ where $||r^k||$ indicates the Euclidean norm of the vector r^k . The τ criteria selects column i to be included in B_k if

$$\tau_i = (J_i^k, F(x^k))^2 > Q$$

for some tolerance Q . Since the tolerance that will be used in this paper for the τ criteria is dependent on the iteration step, it will be denoted by Q_k . In fact, Q_k is the square of the Euclidean norm of the residue vector.

$$Q_k = ||F(x^k)||^2$$

Regardless of the projection subspace sequence that is used to solve a problem, the nonlinear projection method has the same general form. Section II will be concerned with the stepwise development of algorithms for using the nonlinear projection method. These algorithms will be developed independent of the B_k sequence. Section III will be concerned with the stepwise development of algorithms for dynamically determining projection subspaces that take advantage of the criteria discussed above.

II. STEPWISE DEVELOPMENT OF THE NONLINEAR PROJECTION ALGORITHM

In this section, the stepwise development of the nonlinear projection method independent of subspace selection will be considered. There is a twofold goal for this development. First of all, it is hoped that the stepwise development will lead to a solid understanding of the computational requirements of the nonlinear projection method. Secondly, following the successive stepwise refinements, the final algorithm will be in a form that can be easily translated into an arbitrary computer programming language.

In constructing an algorithm, it is necessary at some point to consider the types and structure of the data that will be manipulated. The types and structure of some of the data elements which will be used in this section as well as Section III are defined in the declarations below:

```
declare var  SUBSPACE  :  [1..N] array of integer
declare var  F,X,C,D,DX :  [1..N] array of real
declare var  J,A       :  [1..N,1..N] array of real
declare var  TEMP, NORM, TOL : real
declare var  I,K,M,N,II, KK, DIMENSION : integer
```

The initial algorithm which will consist of only one statement is called algorithm A_0 .

A_0 : begin

solve the N dimensional nonlinear system $F(x) = 0$
using the nonlinear projection method

end

Since the single statement in algorithm A_0 cannot be translated into a computer program very easily, it will be necessary to consider further refinements.

It will be assumed in the remaining discussion that the initial approximation to the solution vector will be an input to the algorithms. Another parameter that will be an input to the algorithms is the dimension of the system being solved, N. If TOL is the accuracy desired in the solution and $\|F\|$ represents the Euclidean norm of F, then the first refinement of A_0 would be:

A_1 : begin

1.1: input an initial approximation to x

1.2: input the dimension of system N

2: while $\|F\| < \text{TOL}$

do begin

calculate a change vector d
update x

end

3: output X, F, $\|F\|$

end

Algorithm A_1 has a structure which could lead to many iterative methods. Particular methods such as the projection method are distinguished by the way in which they "calculate a change vector d ". Algorithm A_2 will introduce more precise actions to replace this statement.

```

A2: begin
1.1:   input an initial approximation to x
1.2:   input the dimension of the system, N
2:     while ||F|| < TOL
        do begin
2.1:       determine the projection subspace
2.2:       solve for dx
2.3:       compute d
2.4:       update x
        end
3:     output X, F, ||F||
end

```

Further refinements of statement 2.1 will be considered in Section III. It will be assumed in this paper, regardless of the method used to select a projection subspace, that the subspace dimension will be indicated by the variable DIMENSION and that SUBSPACE(I), $I = 1, \dots, \text{DIMENSION}$ will indicate which column of the Jacobian matrix will be the I th component of the projection subspace.

In order to obtain dx , it is necessary to construct and solve the appropriate linear system as defined in Chapter 1. Prior to doing that, however, it will be necessary to obtain J which is the value of the Jacobian matrix evaluated at the current approximation to the solution vector as well as F which is the value of the system evaluated at the current approximation to the solution vector. Since J and F may be needed to evaluate the criteria discussed in Chapter 1, it is logical to compute these sets of functions prior to statement 2.1. For convenience, suppose we have a procedure JACOBIAN to compute J and another procedure FUNCTION to compute F . Care must be taken, since depending on the subspace selection criteria which is used, certain other calculations must precede the calculation of new values for J and F . A further refinement of A_2 would be:

```

A3: begin
1.1:   input an initial approximation to x
1.2:   input the dimension of the system N
2:     while ||F|| < TOL
        do begin
                CALL JACOBIAN(J,x)
                CALL FUNCTION(F,x)
2.1:   determine the projection subspace
2.2.1: generate the linear system
        Adx = C

```


2.2.2: solve $A dx = C$ for dx

2.3: compute d

2.4: update x

end

3: output $X, F, ||F||$

end

A further refinement of the statements in algorithm A_3 results in algorithm A_f . Algorithm A_f is in a form which can be easily translated to an arbitrary programming language.

The refinement of statement 2.2.1 takes advantage of the fact that the matrix A is symmetric. Statement 2.2.2 in algorithm A_3 will not be expanded further in this report. It will be assumed that the user of the algorithm has developed a mechanism independent of the projection algorithm to solve linear systems of equations. The refinement of statements 2.3 and 2.4 from algorithm A_3 avoids the explicit use of the vector d .

A_f : begin

1.1: input an initial approximation to x

1.2: input the dimension of the system N

call FUNCTION(F, x)

TEMP = 0

M = 1

Repeat

```

    begin
        TEMP = TEMP + F(M)2
        M = M + 1
    end
until M > N
    NORM = (TEMP)1/2
2: while NORM < TOL
    do begin
        call JACOBIAN(J,x)
2.1: "determine the projection subspace"
2.2.1: I = 1
        Repeat
            begin
                II = SUBSPACE(I)
                K = 1
                Repeat
                    begin
                        KK = SUBSPACE(K)
                        TEMP = 0
                        M = 1
                        Repeat
                            begin
                                TEMP = TEMP + J(M,II)*J(M,KK)
                                M = M + 1
                            end
                    end
                end
            end
    end

```

until M > N

A(I,K) = TEMP

A(K,I) = TEMP

K = K + 1

end

until K > I

TEMP = 0

M = 1

Repeat

begin

TEMP = TEMP - J(M,I)*F(M)

M = M + 1

end

until M > N

C(I) = TEMP

I = I + 1

end

until I > DIMENSION

2.2.2: "solve $Adx = C$ for dx "

2.3: I = 1

Repeat

```

      begin
        II = SUBSPACE(I)
        X(II) = X(II) + DX(I)
        I = I + 1
      end
    until I > DIMENSION
    call FUNCTION(F,X)
    TEMP = 0
    M = 1
    Repeat
      begin
        TEMP = TEMP + F(M)2
        M = M + 1
      end
    until M > N
    NORM = TEMP1/2
  end
3:   output X,F, ||F||
end

```

Now, this algorithm as well as the data structures discussed in the beginning of this chapter could be translated directly to FORTRAN or almost any other algebraic programming language. This translated code can be combined with one of the routines to determine a projection

subspace which will be discussed in Section III and a routine to solve linear systems to obtain a program for solving nonlinear systems of equations with the projection method.

III. STEPWISE DEVELOPMENT OF SUBSPACE SELECTION CRITERIA

In this section, consideration will be given to the refinement of statement 2.1 from algorithm A_2 found in Section II. The initial algorithm A from which the refinement process begins will just consist of this single statement.

```
A: begin
      determine a projection subspace
end
```

Algorithms for the use of the subspace selection criteria described in Section I will be developed in the form of procedures. Each algorithm will be refined from its own initial algorithm which is slightly more explicit than algorithm A. First of all, consider the angle criteria.

```
A0: Procedure ANGLE(J,F,TOLERANCE,SUBSPACE,DIMENSION,N)
      begin
            determine a projection subspace
            using the angle criteria
      end
```

ANGLE is the name of the procedure which uses the angle criteria to determine a projection subspace. For this criteria, J and F, which

are respectively the current values of the Jacobian matrix and the system, will be assumed to be evaluated prior to invoking the procedure. In addition, the value for TOLERANCE will be assumed to be an input parameter from the calling program.

The angle criteria includes column i in the subspace whenever $\cos^2 \theta_{rk_j k_i} > \text{TOLERANCE}$. If $\text{COSINE}(I) = \cos^2 \theta_{rk_j k_i}$, then the first refinement of A_0 would be:

```
A1: Procedure ANGLE(J,F,TOLERANCE,SUBSPACE,DIMENSION,N)
      begin
1:      compute COSINE
2:      determine the values for DIMENSION, SUBSPACE
      end
```

The refinement of statement 2 in algorithm A_1 is very straightforward. In the event that a null subspace is chosen, i.e., $\text{DIMENSION} = 0$, then a subspace which is determined by the n columns of the Jacobian is used by default.

In the refinement of step 1 from algorithm A_1 , FXFX will be a variable whose value is equal to (F,F) .

A_f: Procedure ANGLE(J,F,TOLERANCE,SUBSPACE,DIMENSION,N)

begin

FXFX = 0

I = 1

Repeat

begin

FXFX = FXFX + $F(I)^2$

I = I + 1

end

until I > N

I = 1

while I < N + 1

do begin

TEMP1 = 0

TEMP2 = 0

K = 1

Repeat

begin

TEMP1 = TEMP1 + F(K)*J(K,I)

TEMP2 = TEMP2 + $J(K,I)^2$

K = K + 1

end

```

    until K > N
        COSINE(I) = TEMP12/FXFX•TEMP2
        I = I + 1
    end
    DIMENSION = 0
    I = 1
    Repeat begin
        if COSINE(I) > TOLERANCE
            then begin
                DIMENSION = DIMENSION + 1
                SUBSPACE(DIMENSION) = I
                I = I + 1
            end
            else
                I = I + 1
            end
        until (I > N)
        if DIMENSION = 0
            then
                begin
                    I = 1
                    DIMENSION = N

```



```

      while I < N + 1
        do begin
          SUBSPACE(I) = I
          I = I + 1
        end
      end
    else
      end

```

In a similar manner, an algorithm which determines a projection subspace on the basis of the φ criteria, can be developed. The initial algorithm B_0 would be

B_0 : Procedure PHI(J,F,TOLERANCE,SUBSPACE,DIMENSION,N,FLAG,X)

begin

determine a projection subspace
using the φ criteria

end

Since the φ criteria calculates and uses $\Delta_i^k = J_i^{k+1} - J_i^k$, it is necessary when the φ criteria is being used to explicitly manipulate the Jacobian matrix. Thus, the call to the procedure JACOBIAN should be made within the procedure PHI and not before the procedure PHI is called. Flag is a variable which is used to indicate that the procedure PHI has been called the first time. If Flag = 0, indicating PHI has been called the first time, then

the procedure ANGLE is used to determine the projection subspace.

Let $\text{DELTA}(I) = (\Delta_i^k, \Delta_i^k)$. Then the φ criteria selects column i to be included in the projection subspace at step $k + 1$ when $\text{DELTA}(I) > (J_i^k, J_i^k) \cdot \text{TOLERANCE}$. The following statements are a first refinement of algorithm B_0 .

B_1 : Procedure PHI(J,F,TOLERANCE,SUBSPACE,DIMENSION,N,FLAG,X)

begin

1: if FLAG = 0

then begin

FLAG = 1

call JACOBIAN(J,X)

call ANGLE(J,F,0.25,SUBSPACE,DIMENSION,N)

go to 7

else

2: save J in JOLD

3: evaluate $J(X^{k+1})$

4: compute (J_i^k, J_i^k) $i = 1, 2, \dots, N$

5: compute DELTA

6: determine SUBSPACE

7: end

Note that if this procedure is used, then FLAG must be initialized in the main projection algorithm before the first while statement. I will assume that a value for TOLERANCE will be an input parameter for

the procedure PHI. When the procedure ANGLE is called during the initial call to PHI, a default value for the angle criteria tolerance equal to 0.25 has been assumed.

Further refinements of the steps in algorithm B_1 results in the final algorithm B_f .

B_f : Procedure PHI(J,F,TOLERANCE,SUBSPACE,DIMENSION,N,FLAG,X)

begin

if FLAG = 0

then begin

FLAG = 1

call JACOBIAN(J,X)

call ANGLE(J,F,0.25,SUBSPACE,DIMENSION,N)

go to 7

end

I = 1

while I < N + 1

do begin

K = 1

while K < N + 1

do begin

JOLD(K,I) = J(K,I)

K = K + 1

end

```

        I = I + 1

        end

    call JACOBIAN(J,X)

    I = 1

    Repeat

        begin

            K = 1

            TEMP = 0

            Repeat

                begin

                    TEMP = TEMP + JOLD(K,I)2

                    K = K + 1

                end

            until K > N

            JJ(I) = TEMP

            I = I + 1

        end

    until I > N

    I = 1

    Repeat

        begin

            K = 1

            TEMP = 0

```

```

Repeat
  begin
    TEMP = TEMP + (J(K,I) - JOLD(K,I))2
    K = K + 1
  end
  until K > N
  DELTA(I) = TEMP
  I = I + 1
end
until I > N
DIMENSION = 0
I = 1
while I < N + 1
  do begin
    if DELTA(I) > JJ(I) * TOLERANCE
      then
        begin
          DIMENSION = DIMENSION + 1
          SUBSPACE(DIMENSION) = I
          I = I + 1
        end
      else
        I = I + 1
      end
    end
  end

```

```

if DIMENSION = 0
  then
    begin
      DIMENSION = N
      I = 1
      Repeat
        begin
          SUBSPACE(I) = I
          I = I + 1
        end
      until I > N
    end
  else
    end

```

Algorithm B_f is now in a form that can be translated to a programming language. A similar analysis, for the τ criteria would result in an algorithm which is similar in form to both A_f and B_f .

BIBLIOGRAPHY

1. D. D. Georg, Determining nonlinear projection subspaces, National Technical Information Service, USAEC Report No. IS-3545, 1975.
2. A. MacEachern, A generalized projection method for systems of nonlinear equations, Unpublished Ph.D. thesis, Ames, Iowa, Library, Iowa State University of Science and Technology, 1971.
3. M. White, A class of projection methods for solving systems of n nonlinear equations in n unknowns, allowing projections of order one through n , Unpublished Ph.D. thesis, Ames, Iowa, Library, Iowa State University of Science and Technology, 1971.