*Comp. & Maths. with Appls.* Vol. 12A, No. 10, pp. 1019–1027, 1986
Printed in Great Britain. 
0886-9553/86 $3.00 + 0.00


# AN ITERATIVE ALGORITHM FOR THE SOLUTION OF A TRIDIAGONAL LINEAR SYSTEM OF EQUATIONS

W. S. YOUSIF and D. J. EVANS
Department of Computer Studies, Loughborough University of Technology, Loughborough, Leics., England

(*Received January 1986*)

Communicated by E. Y. Rodin

## 1. INTRODUCTION

In this paper an algorithm is presented to solve the linear system $\mathbb{A}\mathbf{x} = \mathbf{b}$ when the coefficient matrix $\mathbb{A}$ is tridiagonal. The algorithm is based on the Alternating Group Explicit (AGE) iterative method [1].

## 2. DESCRIPTION OF THE AGE METHOD

Consider the tridiagonal system

$$\mathbb{A}\mathbf{x} = \mathbf{b} \tag{1}$$

of order $n$, and let

$$\mathbb{A} = \begin{bmatrix} d_1 & c_1 & & & & \\ a_2 & d_2 & c_2 & & 0 & \\ & a_3 & d_3 & c_3 & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & a_{n-1} & d_{n-1} & c_{n-1} \\ & & & & a_n & d_n \end{bmatrix} \quad \text{with } d_i \geqslant a_i + c_i, \quad i = 1, 2, \ldots, n \text{ and } a_1 = c_n = 0 \tag{2}$$

$$\mathbf{x} = (x_1, x_2, \ldots, x_n)^t$$

and

$$\mathbf{b} = (b_1, b_2, \ldots, b_n)^t.$$

We split the matrix $\mathbb{A}$ into the sum of two matrices,

$$\mathbb{A} = \mathbb{G}_1 + \mathbb{G}_2 \tag{3}$$

where

$$\mathbb{G}_1 = \begin{bmatrix} d_1' & c_1 & & & & \\ a_2 & d_2' & & & & \\ & & d_3' & c_3 & & 0 \\ & & a_4 & d_4' & & \\ & & & & \ddots & \\ & & & & d_{n-1}' & c_{n-1} \\ & & 0 & & a_n & d_n' \end{bmatrix}$$

and

$$\mathbb{G}_2 = \begin{bmatrix} d_1' & & & & & \\ & d_2' & c_2 & & & 0 \\ & a_3 & d_3' & & & \\ & & & \ddots & & \\ & & 0 & & d_{n-2}' & c_{n-2} \\ & & & & a_{n-1} & d_{n-1}' \\ & & & & & d_n' \end{bmatrix} \tag{4a}$$

if $n$ is even, and

$$\mathbb{G}_1 = \begin{bmatrix} d_1' & & & & & \\ & d_2' & c_2 & & & \\ & a_3 & d_3' & & & 0 \\ & & & \ddots & & \\ & 0 & & d_{n-3}' & c_{n-3} & \\ & & & a_{n-2} & d_{n-2}' & \\ & & & & & d_{n-1}' \quad c_{n-1} \\ & & & & & a_n \quad d_n' \end{bmatrix}$$

and

$$\mathbb{G}_2 = \begin{bmatrix} d_1' & c_1 & & & & \\ a_2 & d_2' & & & & \\ & & d_3' & c_3 & & \\ & & a_4 & d_4' & & 0 \\ & & & & \ddots & \\ & 0 & & & d_{n-2}' & c_{n-2} \\ & & & & a_{n-1} & d_{n-1}' \\ & & & & & d_n' \end{bmatrix} \tag{4b}$$

if $n$ is odd, with $d_i' = [d_i/2]$, $i = 1, 2, \ldots, n$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ satisfy the condition $\mathbb{G}_1 + r\mathbb{I}$ and $\mathbb{G}_2 + r\mathbb{I}$ are non-singular for any $r > 0$.

By using equation (3), the matrix equation (1) can now be written in the form

$$(\mathbb{G}_1 + \mathbb{G}_2)\mathbf{x} = \mathbf{b} \tag{5}$$

and, by following a strategy similar to the ADI method [2], $\mathbf{x}^{(k+1/2)}$ and $\mathbf{x}^{(k+1)}$ can be determined implicitly by

$$(\mathbb{G}_1 + r\mathbb{I})\mathbf{x}^{(k+1/2)} = \mathbf{b} - (\mathbb{G}_2 - r\mathbb{I})\mathbf{x}^{(k)}$$

and

$$(\mathbb{G}_2 + r\mathbb{I})\mathbf{x}^{(k+1)} = \mathbf{b} - (\mathbb{G}_1 - r\mathbb{I})\mathbf{x}^{(k+1/2)} \tag{6a}$$

or explicitly by

$$\mathbf{x}^{(k+1/2)} = (\mathbb{G}_1 + r\mathbb{I})^{-1}[\mathbf{b} - (\mathbb{G}_2 - r\mathbb{I})\mathbf{x}^{(k)}]$$

and

$$\mathbf{x}^{(k+1)} = (\mathbb{G}_2 + r\mathbb{I})^{-1}[\mathbf{b} - (\mathbb{G}_1 - r\mathbb{I})\mathbf{x}^{(k+1/2)}], \tag{6b}$$

where $r$ is the iteration parameter given by [3]

$$r = \sqrt{uv} \tag{7}$$

where $u$ and $v$ are the minimum and maximum eigenvalues of the submatrices of $\mathbb{G}_1$ and $\mathbb{G}_2$.

In the cases when the submatrices are singular and the smallest eigenvalue is zero, then the second smallest eigenvalue is considered [4].

Evidently, $(\mathbb{G}_1 + r\mathbb{I})$, $(\mathbb{G}_2 + r\mathbb{I})$, $(\mathbb{G}_1 - r\mathbb{I})$ and $(\mathbb{G}_2 - r\mathbb{I})$ can be determined by inspection with $(\mathbb{G}_1 + r\mathbb{I})$ and $(\mathbb{G}_2 + r\mathbb{I})$ easily invertible. Hence, from equations (6b), if $n$ is even then $\mathbf{x}^{(k+1/2)}$ and $\mathbf{x}^{(k+1)}$ are given by

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}^{(k+1/2)} = \begin{bmatrix} \alpha_2 f_1 & -c_1 f_1 & & & & \\ -a_2 f_1 & \alpha_1 f_1 & & & & \\ & & \alpha_4 f_3 & -c_3 f_3 & & \\ & & -a_4 f_3 & \alpha_3 f_3 & & \\ & & & & \ddots & \\ & & & & \alpha_n f_{n-1} & -c_{n-1} f_{n-1} \\ & & & & -a_n f_{n-1} & \alpha_{n-1} f_{n-1} \end{bmatrix}$$

$$\times \begin{bmatrix} b_1 - \beta_1 x_1 \\ b_2 - \beta_2 x_2 - c_2 x_3 \\ b_3 - a_3 x_2 - \beta_3 x_3 \\ b_4 - \beta_4 x_4 - c_4 x_5 \\ \vdots \\ b_{n-1} - a_{n-1} x_{n-2} - \beta_{n-1} x_{n-1} \\ b_n - \beta_n x_n \end{bmatrix}^{(k)} \tag{8a}$$

and

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix}^{(k+1)} = \begin{bmatrix} 1/\alpha_1 & & & & & \\ & \alpha_3 f_2 & -c_2 f_2 & & & \\ & -a_3 f_2 & \alpha_2 f_2 & & & \\ & & & \ddots & & \\ & & & & \alpha_{n-1} f_{n-2} & -c_{n-2} f_{n-2} \\ & & & & -a_{n-1} f_{n-2} & \alpha_{n-2} f_{n-2} \\ & & & & & 1/\alpha_n \end{bmatrix}$$

$$\times \begin{bmatrix} b_1 - \beta_1 x_1 - c_1 x_2 \\ b_2 - a_2 x_1 - \beta_2 x_2 \\ b_3 - \beta_3 x_3 - c_3 x_4 \\ \vdots \\ b_{n-2} - a_{n-2} x_{n-3} - \beta_{n-2} x_{n-2} \\ b_{n-1} - \beta_{n-1} x_{n-1} - c_{n-1} x_n \\ b_n - a_n x_{n-1} - \beta_n x_n \end{bmatrix}^{(k+1/2)}, \tag{8b}$$

where

$$\alpha_i = d_i' + r \quad \text{and} \quad \beta_i = d_i' - r, \quad i = 1, 2, \ldots, n$$

and

$$f_i = 1/(\alpha_i \alpha_{i+1} - a_{i+1} c_i), \quad i = 1, 2, \ldots, n-1. \tag{9}$$

By carrying out the necessary algebra, equations (8a, b) can then be written in explicit form:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}^{(k+1/2)} = \begin{Bmatrix} [\alpha_2(b_1 - \beta_1 x_1) - c_1(b_2 - \beta_2 x_2 - c_2 x_3)] f_1 \\ [-a_2(b_1 - \beta_1 x_1) + \alpha_1(b_2 - \beta_2 x_2 - c_2 x_3)] f_1 \\ [\alpha_4(b_3 - a_3 x_2 - \beta_3 x_3) - c_3(b_4 - \beta_4 x_4 - c_4 x_5)] f_3 \\ [-a_4(b_3 - a_3 x_2 - \beta_3 x_3) + \alpha_3(b_4 - \beta_4 x_4 - c_4 x_5)] f_3 \\ \vdots \\ [\alpha_n(b_{n-1} - a_{n-1} x_{n-2} - \beta_{n-1} x_{n-1}) - c_{n-1}(b_n - \beta_n x_n)] f_{n-1} \\ [-a_n(b_{n-1} - a_{n-1} x_{n-2} - \beta_{n-1} x_{n-1}) + \alpha_{n-1}(b_n - \beta_n x_n)] f_{n-1} \end{Bmatrix}^{(k)} \tag{10a}$$

and

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix}^{(k+1)} = \begin{Bmatrix} (b_1 - \beta_1 x_1 - c_1 x_2)/\alpha_1 \\ [\alpha_3(b_2 - a_2 x_1 - \beta_2 x_2) - c_2(b_3 - \beta_3 x_3 - c_3 x_4)]f_2 \\ [-a_3(b_2 - a_2 x_1 - \beta_2 x_2) + \alpha_2(b_3 - \beta_3 x_3 - c_3 x_4)]f_2 \\ \vdots \\ [\alpha_{n-1}(b_{n-2} - a_{n-2}x_{n-3} - \beta_{n-2}x_{n-2}) \\ \qquad - c_{n-2}(b_{n-1} - \beta_{n-1}x_{n-1} - c_{n-1}x_n)]f_{n-2} \\ [-a_{n-1}(b_{n-2} - a_{n-2}x_{n-3} - \beta_{n-2}x_{n-2}) \\ \qquad + \alpha_{n-2}(b_{n-1} - \beta_{n-1}x_{n-1} - c_{n-1}x_n)]f_{n-2} \\ (b_n - a_n x_{n-1} - \beta_n x_n)/\alpha_n \end{Bmatrix}^{(k+1/2)}. \quad (10b)$$

Equations (10a) can be derived from the computational molecule shown in Fig. 1 and the equations

$$x_i^{(k+1/2)} = A_i x_{i-1}^{(k)} + B_i x_i^{(k)} + C_i x_{i+1}^{(k)} + D_i x_{i+2}^{(k)} + E_i \quad (11a)$$

and

$$x_{i+1}^{(k+1/2)} = \bar{A}_{i+1} x_{i-1}^{(k)} + \bar{B}_{i+1} x_i^{(k)} + \bar{C}_{i+1} x_{i+1}^{(k)} + \bar{D}_{i+1} x_{i+2}^{(k)} + \bar{E}_{i+1}, \quad (11b)$$

where

$$A_i = -\alpha_{i+1} a_i f_i, \qquad B_i = -\alpha_{i+1}\beta_i f_i, \qquad C_i = c_i \beta_{i+1} f_i,$$
$$D_i = c_i c_{i+1} f_i \quad \text{and} \quad E_i = (\alpha_{i+1} b_i - c_i b_{i+1}) f_i \quad (12a)$$

and

$$\bar{A}_{i+1} = a_i a_{i+1} f_i, \qquad \bar{B}_{i+1} = a_{i+1}\beta_i f_i, \qquad \bar{C}_{i-1} = -\alpha_i \beta_{i+1} f_i,$$
$$\bar{D}_{i+1} = -\alpha_i c_{i+1} f_i \quad \text{and} \quad \bar{E}_{i+1} = (-a_{i+1} b_i + \alpha_i b_{i+1}) f_i \quad (12b)$$

for $i = 1, 3, 5, \ldots, n-1$ with $a_1 = c_n = 0$. Similarly the explicit computational molecule for the $(k+1)$th sweep given by equations (10b) is given in Fig. 2 and by the equations

$$x_i^{(k+1)} = P_i x_{i-1}^{(k+1/2)} + Q_i x_i^{(k+1/2)} + R_i x_{i+1}^{(k+1/2)} + S_i x_i^{(k+1/2)} + T_i \quad (13a)$$

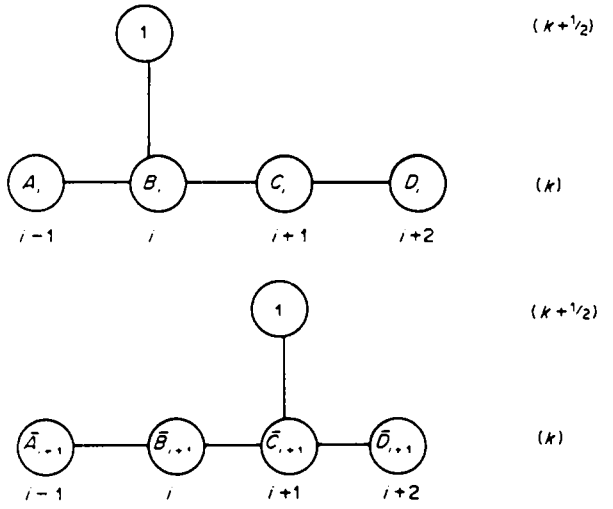


Fig. 1. Explicit computational molecule for the $(k+\frac{1}{2})$th sweep.

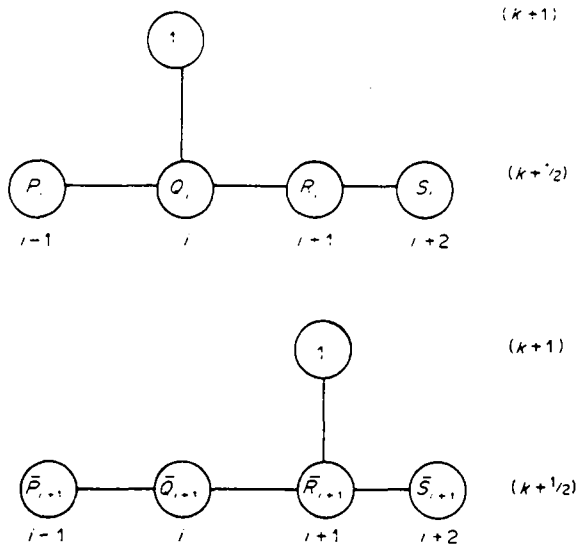


Fig. 2. Explicit computational molecule for the $(k+1)$th sweep.

and

$$x_{i+1}^{(k+1)} = \bar{P}_{i-1}x_{i-1}^{(k+1.2)} + \bar{Q}_{i+1}x_i^{(k+1/2)} + \bar{R}_{i+1}x_{i-1}^{(k+1.2)} + \bar{S}_{i+1}x_i^{(k+1.2)} + \bar{T}_{i+1}, \tag{13b}$$

where

$$P_i = -\alpha_{i+1}a_i f_i, \quad Q_i = -\alpha_{i+1}\beta_i f_i, \quad R_i = c_i\beta_{i+1}f_i,$$
$$S_i = c_i c_{i+1} f_i \quad \text{and} \quad T_i = (\alpha_{i+1}b_i - c_i b_{i+1})f_i \tag{14a}$$

and

$$\bar{P}_{i+1} = a_i a_{i+1} f_i, \quad \bar{Q}_{i+1} = a_{i+1}\beta_i f_i, \quad \bar{R}_{i+1} = -\alpha_i\beta_{i+1}f_i,$$
$$\bar{S}_{i+1} = -\alpha_i c_{i+1} f_i \quad \text{and} \quad \bar{T}_{i+1} = (-a_{i+1}b_i + \alpha_i b_{i+1})f_i \tag{14b}$$

for $i = 2, 4, \ldots, n-2$; with $i = 0$,

$$\bar{P}_1 = 0, \quad \bar{Q}_1 = 0, \quad \bar{R}_1 = -\beta_1/\alpha_1, \quad \bar{S}_1 = -c_1/\alpha_1 \quad \text{and} \quad \bar{T}_1 = b_1/\alpha_1$$

and $i = n$,

$$P_n = -a_n/\alpha_n, \quad Q_n = -\beta_n/\alpha_n, \quad R_n = 0, \quad S_n = 0 \quad \text{and} \quad T_n = b_n/\alpha_n.$$

In a similar manner, a similar set of equations can be obtained if $n$ is odd. Thus the AGE algorithm can be completed explicitly by using equations (11a, b) and (13a, b) in alternate sweeps until a suitable convergence to a specific level of accuracy $\epsilon$ is achieved.

## 3. NUMERICAL RESULTS

Solve the linear system,

$$\begin{aligned} 4x_1 - x_2 &= b_1 \\ -x_{i-1} + 4x_i - x_{i+1} &= b_i \quad i = 2, 3, \ldots, n-1 \\ -x_{n-1} + 4x_n &= b_n \end{aligned} \tag{15}$$

when the constant vector **b** is given random values.

Table 1

| Order of matrix $n$ | Iteration parameter $r$ | No. of iterations $k$ |
|---|---|---|
| 10 | 1.43–1.96 | 5 |
| 20 | 1.43–1.777 | 5 |
| 30 | 1.43–1.776 | 5 |
| 40 | 1.47–1.739 | 5 |

A Fortran program which solves this problem using the subroutine (AGE) is given in the Appendix and Table 1 shows the results obtained, the convergence test used was the absolute test $|x_i^{(k-1)} - x_i^{(k)}| < \epsilon$, with $\epsilon = 10^{-4}$.

## 4. THE COMPUTATIONAL COMPLEXITY OF THE ALGORITHM

The computational complexity for the sequential algorithm can be easily shown from the above method to be 8 multiplications and 8 additions per point per AGE iteration with an additional 2 multiplications and 3 additions before the first iteration for the evaluation of $f$.

Hence, for the above (model) problem, the total number of arithmetic operations required for a solution is

$$40n + 2 \text{ multiplications and } 40n + 3 \text{ additions} \tag{16}$$

However, it is well-known that a direct tridiagonal system solver applied to the given linear system (15) requires [5]

$$5n \text{ multiplications and } 3n \text{ additions} \tag{17}$$

Further, for non-linear problems, it is not so easy to apply direct methods and the application of the AGE iterative algorithm is of immense practical importance. Also, the AGE method becomes extremely competitive if a good previous approximation is used as a starting guess, i.e. as in parabolic problems in which case only 2–3 iterations are required to produce the solution.

For parallel computers, it is well-known that the direct method consists of non-linear recurrences that must be evaluated sequentially, so there is little parallelism in the direct algorithm and therefore it is not an ideal algorithm for use on parallel computers.

However, the AGE algorithm is suitable for parallel computers as it possesses separate and independent tasks, i.e. $(2 \times 2)$ groups which can be executed concurrently. Thus, from equations (10a, b) the total number of arithmetic operations required for a solution on a synchronous SIMD computer can be verified to be 16 multiplications and 16 additions per iteration if $n/2$ processors are available taking all the odd points followed by the even points, and on asynchronous MIMD computers it requires 8 multiplications and 8 additions per iteration if $n$ processors are available.

## REFERENCES

1. D. J. Evans and W. S. Yousif, The Alternating Group Explicit (A.G.E.) method for two point boundary value problems. *Proc. IMACS 11th Wld Congr.*, Oslo, Vol. 1 (Edited by R. Vichnevetsky and J. Vignes), pp. 213–220. North-Holland, Amsterdam (1985).
2. D. W. Peaceman and H. H. Rachford Jr, The numerical solution of parabolic and elliptic differential equations. *J. Soc. ind. appl. Math.* **3**, 28–41 (1955).
3. D. J. Evans, Group explicit iterative methods for solving large linear systems. *Int. J. Comput. Math.* **17**, 81–108 (1985).
4. R. B. Kellogg and J. Spanier, On optimal alternating direction parameters for singular matrices. *Maths. Comput.* **19**, 448–452 (1965).
5. R. S. Varga, *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, N.J. (1962).

*(See overleaf for the Appendix)*

# APPENDIX

```
c
c
c
c
c
c  **** To solve a system  Ax=b  when A is a tridiagonal ****
c  ************    matrix by the (a.g.e.)method.    *********
c
c     age   - Tridiagonal equation solver.
c         n   the order of the system
c         m   the iteration counter
c         a   the sub diagonal of A
c         c   the super diagonal of A
c         d   half the diagonal of A
c         b   the constant vector
c
c

      dimension a(41),b(41),c(41),d(41),x(41)
c
      read(5,*)r,n,eps
      do 10 i=1,n
        a(i)=-1.
        d(i)=2.
        c(i)=-1.
        b(i)=rand()*10.
10    continue
      a(1)=0.
      c(n)=0.
      write(6,20)(b(i),i=1,n)
c
      call age(a,b,c,d,m,n,eps,x,r)
c
      write(6,15)n,eps,m,r
      write(6,20)(x(i),i=1,n)
15    format(/5x,'n =',i3,5x,'eps =',f9.6,
     1 /5x,'No. of iterations =',i4,5x,'r =',f7.3/5x,
     2 'The solution vector is'/5x,22('"'))
20    format(1x,10f8.4)
      stop
      end
c
c
      subroutine age(a,b,c,d,m,n,eps,x,r)
      dimension a(41),b(41),c(41),d(41),x(41),x1(0:41),x2(0:41),f(41)
      n1=n-1
      n2=n-2
      do 25 i=1,n+1
      x(i)=0.1
      f(i)=1./((d(i)+r)*(d(i+1)+r)-a(i+1)*c(i))
25    continue
      m=0
30    continue
      m=m+1
      do 35 i=1,n
35    x1(i)=x(i)
```

```
c   **** first sweep ****
c        ''''''''''''''''''
       if(mod(n,2).eq.0)go to 40
        x2(1)=(b(1)-(d(1)-r)*x1(1)-c(1)*x1(2))/(d(1)+r)
        j=2
       go to 45
40      j=1
45     do 50 i=j,n1,2
        i1=i+1
        i2=i+2
        k=i-1
        d1=d(i)+r
        d2=d(i1)+r
       p1=b(i)-a(i)*x1(k)-(d(i)-r)*x1(i)
       p2=b(i1)-(d(i1)-r)*x1(i1)-c(i1)*x1(i2)
       x2(i)=(d2*p1-c(i)*p2)*f(i)
       x2(i1)=(-a(i1)*p1+d1*p2)*f(i)
50     continue

c  **** second sweep ****
c       ''''''''''''''''''''
       if(mod(n,2).ne.0)go to 55
        x(1)=(b(1)-(d(1)-r)*x2(1)-c(1)*x2(2))/(d(1)+r)
        j=2
       go to 60
55      j=1
60     do 65 i=j,n2,2
         i1=i+1
         i2=i+2
         k=i-1
         d1=d(i)+r
         d2=d(i1)+r
        p1=b(i)-a(i)*x2(k)-(d(i)-r)*x2(i)
        p2=b(i1)-(d(i1)-r)*x2(i1)-c(i1)*x2(i2)
       x(i)=(d2*p1-c(i)*p2)*f(i)
       x(i1)=(-a(i1)*p1+d1*p2)*f(i)
65     continue
       x(n)=(b(n)-a(n)*x2(n1)-(d(n)-r)*x2(n))/(d(n)+r)
       do 70 i=1,n
 70     if(abs(x(i)-x1(i)).gt.eps)go to 30
       return
       end
```