数理解析研究所講究録
第 746 巻 1991 年 140-145

LAMAX-S
A Language for Matrix Calculation on Super Computers

Satoshi Uchida
Research Associate
Faculty of Eng.
Kanagawa University

Yutaka Karasawa
Prof.& Dr.of Eng.
Faculty of Eng.
Kanagawa University

Naokazu Yamaki
Research Institute
of Systems Planning

Shigeru Hongo
Associate Prof.
Business Administration
Senshu University

January 31, 1991

1 Introduction

In supercomputing, matrix calculation and vector calculation are used frequently. But because of the lack of matrix calculation function in FORTRAN, it is very difficult to write an efficient program for various super computers. In most cases, we have to tune up a program so as to fit the program coding to a peculiarity of each super computer.

We have been designed and implemented a new programming language, LAMAX-S (a LAnguage for MAtriX calculation on super computer) and its processor. LAMAX-S is an enhanced FORTRAN:

$$\boxed{\text{LAMAX-S}} = \boxed{\text{FORTRAN}} + \boxed{\text{Matrix class and its calculation}}$$

The purpose of LAMAX-S is as follows.

writability & readability To introduce matrix class to FORTRAN, we can write a program easily which uses a matrix calculation. Matrix calculation can be programmed as an expression directly. And there are many operations for matrix data.

efficiency LAMAX-S is designed so as to run as fast as possible on various super computers. LAMAX-S generates a well-tuned FORTRAN source program for various super computers.

LAMAX-S is realized as a preprocessor to FORTRAN. That is, LAMAX-S source program is translated to the equivalent FORTRAN source program by the LAMAX-S preprocessor. Then, the FORTRAN program is compiled by FORTRAN compiler, and linked with Matrix calculation library. This process is shown in Fig. 1.

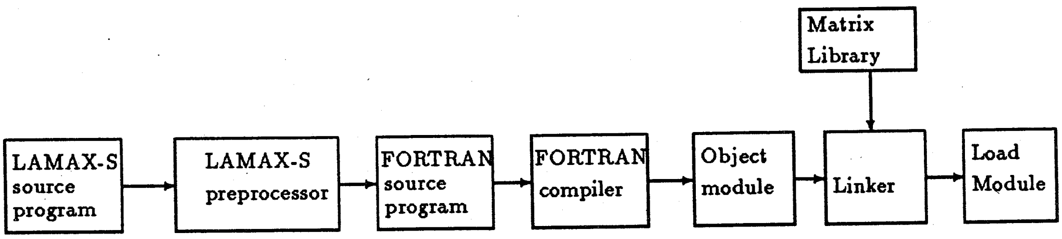

Fig. 1. The flow of LAMAX-S system

In this paper, we will introduce the outline of LAMAX-S informally.

2 A brief introduction using sample programs

2.1 Regression

In order to show the image of LAMAX-S, we show a brief sample program list in program 1. This program is for calculating regression model. In the following regression model,

$$Y = X\beta + \varepsilon$$

where, Y is a vector which has n elements, X is $n \times k$ matrix, β is a vector which has k elements, ε is a vector which has n elements. $\widehat{\beta}$ is definded as follows:

$$\widehat{\beta} = (X'X)^{-1}X'Y$$

```
 1 c
 2 c      Regression
 3 c
 4        parameter(n=10,k=5)
 5        real*8 X:rectangluar[n,k]
 6        real*8 Y:vector[n]
 7        real*8 B:vector[k]
 8        real*8 E:vector[n]
 9
10        call minput(X)
11        call minput(Y)
12
13        B = 1 / (X'*X) * X' * Y
14        E = Y - X*B
15
16        call mprint(B)
17        call mprint(E)
18
19        stop
20        end
```

program 1. The regression program

As LAMAX-S is an enhanced FORTRAN, most of FORTRAN statements can be included in a LAMAX-S source program. Line 1-3 are comment line. Line 4,19 and 20 are all FORTRAN statements. Line 5-8 declare matrix and vector variables. In Line 5, X is declared as an n $\times$ k rectangular matrix variable which element type is real*8 (double precision). Y and E are both vector variables which have n elements, and which element type is also real*8. B is also a vector variable which has k elements. Line 10-11, 16 and 17 are LAMAX-S built-in subroutine call. A built-in subroutine named "minput" inputs matrix or vector data in a special format. Also "mprint" subroutine prints out matrix or vector data in a special format. Line 13,14 are expression which calculate the regression. In LAMAX-S, a representation of an inverse matrix uses a division expression: for example, an inverse matrix A or A^{-1} is represented as 1/A. An operator '(single quotation) represents a transposed matrix. Therefore, an expression,

$$\widehat{\beta} = (X'X)^{-1}X'Y$$

is represented in LAMAX-S as follows:

```
B = 1 / (X'X) * X' * Y
```

2.2 simultaneous linear equation using Jacobi-method

Program 2. shows a program of simultaneous linear equation using Jacobi-method. We solve the following simultaneous linear equation:

$$AX = b$$

In the first step of the method, matrix A is split as follows:

$$A = L + D + U$$

where L is a lower triangular matrix, D is a diagonal matrix, U is an upper triangular matrix. Using the following expression, an approximate value is renewal.

$$X_{(new)} = D^{-1}(b - (L + U)X_{(old)})$$

In line 18, matrix A is split to L (lower triangular part of A), D (diagonal part of A), and U (lower triangular part of A). In line 20, `0::vector[n]` means a constant vector which has n elements and all the values are zero. In line 24, "absmax" function is a build-in function of LAMAX-S. The function returns a maximum absolute value which absolute value is a maximum in the argument.

```
c
c     simultaneous linear equation using Jacobi-method
c
      parameter(n=100, eps=1D-12)

      real*8 A:square[n]

      real*8 D:diagonal[n]
      real*8 L:lower_tr[n,1]
      real*8 U:upper_tr[n,1]
      real*8 X:vector[n]
      real*8 NewX:vector[n]
      real*8 B:vector[n]

      call minput(A)
      call minput(B)

      | L ^ D ^ U |{-1,1} << A

      X = 0::vector[n]

      do 10 i=1,100
           NewX = 1/D * ( B - ( L + U ) * X )
           if(absmax(NewX-X).le.eps) goto 20
           X = NewX
   10 continue

       write(*,*) 'could not solve after 100 iterations'
       goto 30

   20 continue
       write(*,*) 'could solve'
       call mprint(NewX)

   30 continue
       stop
       end
```

Program 2. simultaneous linear equation using Jacobi-method

3 LAMAX-S grammar

3.1 Matrix declaration

Table 1. shows the matrix form in LAMAX-S. As shown in Table 1., each matrix/vector variable has three attributes: form, density and symmetry. The underlined attribute is default.

Table 1. The Form in LAMAX-S

Form	Density of element	Symmetry
rectangular(m×n) matrix square matrix vector row vector band matrix diagonal matrix tridiagonal matrix upper triangular matrix lower triangular matrix	<u>dense</u> sparse	symmetric <u>asymmetric</u>

These attributes are declared in declaration statements. In LAMAX-S, the form of declaration statement is as follows.

Table 2. Declaration of Form

explanation of matrix form	syntax of declaration
rectangular(m×n) matrix	`rectangular[m,n]`
(m×m) square matrix	`square[m]`
vector which has m elements	`vector[m]`
row vector which has m elements	`rvector[m]`
(m×m, s upper band wide, t lower band wide) band matrix	`band[m,s,t]`
(m×m) diagonal matrix	`diagonal[m]`
(m×m) tridiagonal matrix	`diag_3[m]`
(m×m) upper triangle matrix	`upper_tr[m]`
(m×m) lower triangle matrix	`lower_tr[m]`

Density of element and symmetricity are specified after size parameter followed by colon (:) in a type declaration statement. For example, the following statement declares A as a rectangular matrix which size is 100 × 10, and its elements are all integer.

```
integer A:rectangular[100,10]
```

The following statement declares A as a square matrix which size is 100 × 100, and also which is symmetric.

```
integer A:square[100:symmetric]
```

And the following statement declares A as a sparse matrix.

```
integer A:rectangular[100,10:sparse[150]]
```

"sparse[150]" indicates that the matrix A has 1,000 elements, but as the matrix is sparse, the actual non zero elements are at most 150.

Of course, the following statement declares a symmetric sparse matrix.

```
integer A:square[100:symmetric,sparse[150]]
```

Now several examples are shown below:

declaration in a program	meaning	element type
`integer A:rectangular[m,n]`	m×n rectangular matrix	integer
`real*8 A:band[m,nu,nl]`	m×m band matrix	real*8
`complex A:band[1000,nu,ul:sparse[200]]`	1000×1000 band matrix	complex

3.2 Matrix operator

For the writability, LAMAX-S has an enough operator to write a complex matrix calculation. Table 3. shows the matrix calculation operators.

Table 3. Matrix calculation operators in LAMAX-S [1]

priority	operator	meaning	syntax	result form
1	'	transposed	M'	M
2	!	LU decomposition	!M	M
3	**	power	S**S,M**S	S,M
4	*	matrix multiplication	S*S,S*M,M*S,M*M	S,M,M,M
4	&	multiplication by element	M&M	M
4	/	division	S/S, M/S	S,M
4	/	inverse matrix	S/M, M/M	M,M
4	%	division by element	M%M	M
4	?	solution	M?M	M
4	//	concatenation	C//C	C
5	+	positive sign	+S,+M	S,M
5	−	negative sign	−S,−M	S,M
5	+	addition	S+S,M+M	S,M
5	−	subtraction	S−S,M−M	S,M
6	=	assignment	S=S,M=M	S,M
6	:=	enforced assignment	M:=M	
6	<=	insertion	M{S,S}<=M	
6	=>	extraction	M{S,S}=>M	
6	<=>	exchange	M<=>M	
6	<<	partition	M<<M	

4 Actual implementation

The LAMAX-S precompiler processor has been implemented by **ISP**(Research Institute of Systems Planning.) The latest processor is called as version 1.1. Version 1.1 is only for **PIAX** machine, which is a personal super computer developed by **NKK.** Now, we are planning to implement the processor for other super computers.

It is very sorry that version 1.1 lacks the following two functions.

- sparse matrix calculation
- optimization peculiar to matrix calculation.

As a matrix library, version 1.1 uses LINPACK. Namely, LAMAX-S processor generates FORTRAN program which include LINPACK subroutine call. But in other super computer, we are going to use a built-in libraries of the super computer. For example, in NEC SX, ASL will be used as a library. In Hitachi S-820, Matrix/HAP will be used.

[1] `S:scalar, M:matrix or vector, C:character`

The first step of our project, that is, to implement usable processor of LAMAX-S will be accomplished. Our Next step consists of the following three:

- to introduce sparse matrix calculation
- to introduce optimization peculiar to matrix calculation.
- to support other super computer