Introdução ao Fortran 90 - 3

Alexandre Diehl

Departamento de Física – UFPel

Identificador na forma de Matriz

Definição 1

Um identificador na forma de uma matriz consiste de um conjunto retangular de elementos, todos do mesmo tipo e espécie do tipo, também chamados de variáveis compostas homogêneas.

Definição 2

Uma matriz é um grupo de posições na memória do computador, as quais são acessadas por intermédio de um único nome, fazendo-se uso dos subscritos da matriz.

- Uma matriz de dimensão 1 é chamada de vetor.
- Uma matriz pode ter até 7 subscritos, cada um relacionado com uma dimensão da matriz.

Identificador na forma de Matriz

Definição 1

Um identificador na forma de uma matriz consiste de um conjunto retangular de elementos, todos do mesmo tipo e espécie do tipo, também chamados de arranjos.

Definição 2

Uma matriz é um grupo de posições na memória do computador, as quais são acessadas por intermédio de um único nome, fazendo-se uso dos subscritos da matriz.

- Os índices de cada subscrito da matriz são constantes inteiras, que começam em 1 ou a partir de um intervalo fornecido.

Declaração de Matrizes - Alocação Fixa

Vetor (Dimensão 1)

```
INTEGER, PARAMETER :: N = 100
REAL, DIMENSION (N) :: B
REAL, DIMENSION(10) :: A
INTEGER, DIMENSION(0:9) :: C
END
```

- Posto (*rank*): número de dimensões da matriz. Escalar (posto 0), vetor (posto 1), matriz (posto ≥ 2)
- Extensão (*extent*): número de componentes da cada dimensão da matriz
- Forma (*shape*): vetor cujos componentes são a extensão de cada dimensão da matriz.
- Tamanho (*size*): número total de elementos da matriz.

Declaração de Matrizes - Alocação Fixa

Matriz (Dimensão 2)

```
INTEGER, PARAMETER :: N = 2, M = 5
REAL, DIMENSION (N,M) :: C
REAL, DIMENSION (0:N,0:M) :: D
REAL, DIMENSION(-3:4,7) :: A
INTEGER, DIMENSION(8,0:8) :: B
END
```

Matriz A

- Posto (*rank*): 2
- Extensão (*extent*): 8 e 7
- Forma (*shape*): (/8,7/)
- Tamanho (*size*): 56

Matriz B

- Posto (*rank*): 2
- Extensão (*extent*): 8 e 9
- Forma (*shape*): (/8,9/)
- Tamanho (*size*): 72

Declaração de Matrizes - Alocação Fixa

```
CHARACTER(LEN=25), DIMENSION(5) :: nome
nome(1) = 'Joao Victor'
nome(2) = 'Maria Joana'
nome(3) = 'Alexandre Diehl'
nome(4) = 'Bruno Duarte'
nome(5) = 'Vinicius Silva'
END
```

- Vetor nome, com posto 1 e tamanho 5:

  nome(1) nome(2) nome(3) nome(4) nome(5)

- Cada elemento do vetor nome é do tipo CHARACTER com até 25 caracteres.

Construtores de Matrizes - Vetores

- Usados para inicializar os elementos de um vetor
- Forma geral do construtor: (/ < lista de valores> /)

```
implicit none
real, dimension(4) :: A = (/1.0,3.0,-1.5,4.5/)
integer,dimension(3), parameter :: B = (/1,4,6/)
real(8), dimension(4) :: C
C = (/1.0d0, 3.0d0, -1.5d0, 4.5d0/)
end
```

- Vetor A (variável), com posto 1 e tamanho 4:

  A(1) = 1.0 A(2) = 3.0 A(3) = -1.5 A(4) = 4.5

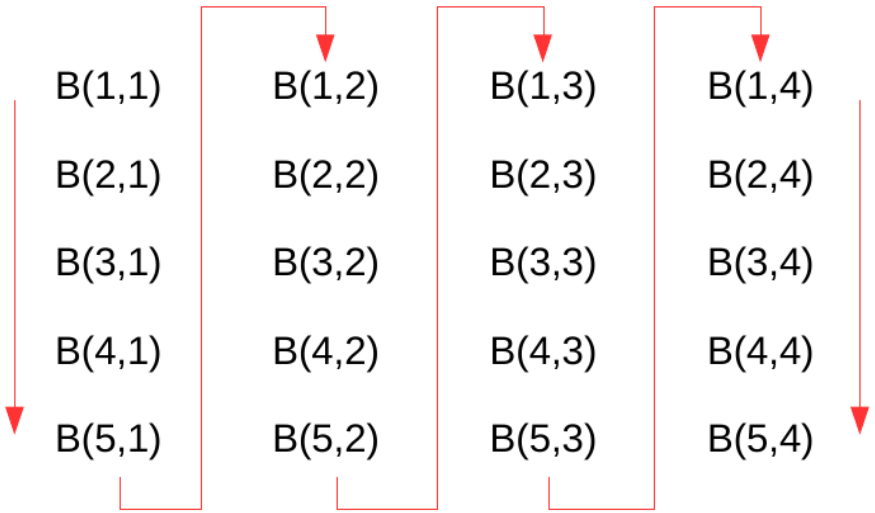
- Vetor B (constante), com posto 1 e tamanho 3.

  B(1) = 1 B(2) = 4 B(3) = 6

Matriz multidimensional: ordem dos elementos

O ordenamento é feito variando primeiro o índice da primeira dimensão, depois da segunda dimensão e assim por diante.

$$B = \begin{pmatrix} 1 & 4 & 0 & 10 \\ -2 & 4 & 3 & 0 \\ 5 & 3 & 0 & 11 \\ 7 & 1 & 2 & 9 \\ 11 & 1 & 4 & 4 \end{pmatrix}$$

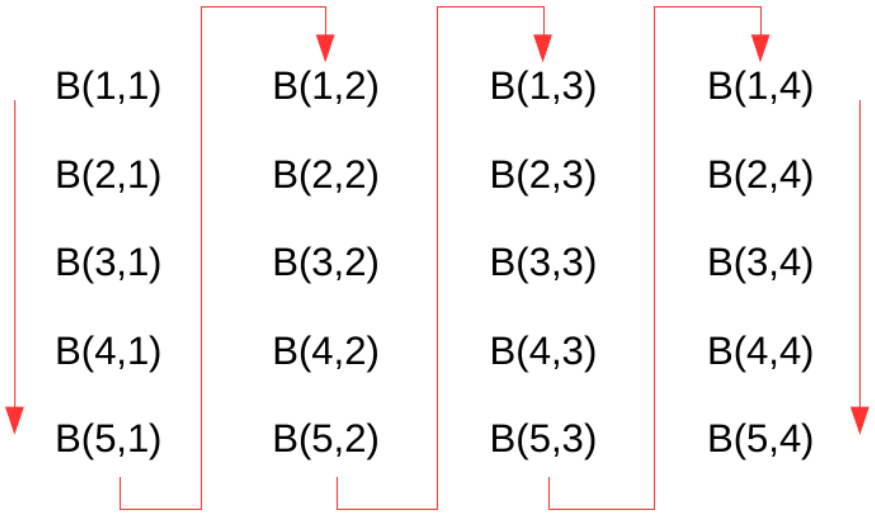


- Os arranjos (matrizes) multidimensionais são organizados por colunas
- O acesso aos elementos de uma matriz multidimensional é mais rápido pelas suas colunas

Matriz multidimensional: ordem dos elementos

```
INTEGER, DIMENSION(5,4) :: B
B(1,1) = 1 ; B(1,2) = 4; B(1,3) = 0; B(1,4) = 10
B(2,1) = -2; B(2,2) = 4; B(2,3) = 3; B(2,4) = 0
B(3,1) = 5 ; B(3,2) = 3; B(3,3) = 0; B(3,4) = 11
B(4,1) = 7 ; B(4,2) = 1; B(4,3) = 2; B(4,4) = 9
B(5,1) = 11; B(5,2) = 1; B(5,3) = 4; B(5,4) = 4
print*,B
END
```

$$B = \begin{pmatrix} 1 & 4 & 0 & 10 \\ -2 & 4 & 3 & 0 \\ 5 & 3 & 0 & 11 \\ 7 & 1 & 2 & 9 \\ 11 & 1 & 4 & 4 \end{pmatrix}$$



Construtores de Matrizes Multidimensionais

```
INTEGER, DIMENSION(2,2) :: A
REAL, DIMENSION(2,3) :: B
A = reshape((/1.0,0.0,0.0,1.0/),shape=(/2,2/))
B = reshape(source=(/1,-2,4,4,0,3/), shape=(/2,3/))
print*,A
print*,B
END
```

- Função Intrínseca SHAPE(SOURCE):
  retorna um vetor inteiro padrão, contendo a forma da matriz ou escalar SOURCE
- Função Intrínseca RESHAPE(SOURCE,SHAPE):
  retorna uma matriz com forma dada pelo vetor inteiro SHAPE e tipo e espécie iguais aos da matriz SOURCE

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 4 & 0 \\ -2 & 4 & 3 \end{pmatrix}$$

Construtores de Matrizes Multidimensionais

```
REAL, DIMENSION(2,3) :: B
B = reshape(source=(/1,-2,4,4,0,3/),shape=(/2,3/),order=(/2,1/))
print*,B
END
```

- Função Intrínseca RESHAPE(SOURCE,SHAPE,ORDER):

  retorna uma matriz com forma dada pelo vetor inteiro SHAPE e tipo e espécie iguais aos da matriz SOURCE, armazenados de acordo com a ordem das dimensões listadas no arrranjo ORDER (opcional).

$$B = \begin{pmatrix} 1 & -2 & 4 \\ 4 & 0 & 3 \end{pmatrix}$$

Alocação Dinâmica de Matrizes

Definição 1

Usada quando não queremos usar alocação fixa para matrizes durante a etapa de compilação do código-fonte.

Definição 2

Permite que a extensão, forma e tamanho, ou seja, o espaço de memória utilizado, da matriz seja alocado durante a execução do código-fonte.

Definição 3

Permite que o espaço de memória alocado à matriz seja liberado durante a execução do código-fonte.

Exige o uso de dois comandos

- No campo de declaração usamos o atributo ALLOCATABLE para declarar o posto da matriz
- O posto (número de dimensões) é indicado pelo número de símbolos : separados por vírgula
- No campo de execução, uma vez conhecido o tamanho da matriz, alocamos o espaço de memória usando a comando ALLOCATE

```
REAL, DIMENSION(:), ALLOCATABLE :: A
INTEGER, DIMENSION(:,:), ALLOCATABLE :: B
INTEGER :: N, M
READ*,N, M
ALLOCATE (A(N))
ALLOCATE (B(N,M))
END
```

→ Matriz A: posto 1, tamanho N → Matriz B: posto 2, forma (/N,M/), tamanho N*M

Liberando o espaço de memória utilizado

- Não sendo mais necessárias, os espaços de memória utilizados pelas matrizes podem ser liberados usando o comando DEALLOCATE
- Sem o comando DEALLOCATE os espaços de memória serão liberados na finalização da execução do código (comando END)

```
REAL, DIMENSION(:), ALLOCATABLE :: A
INTEGER, DIMENSION(:,:), ALLOCATABLE :: B
INTEGER :: N, M
READ*,N, M
ALLOCATE (A(N))
ALLOCATE (B(N,M))
! bloco de comandos
DEALLOCATE (A)
DEALLOCATE (B)
END
```

Operações com Matrizes

- No Fortran 90 as matrizes são tratadas como um objeto único
- Para que as operações envolvendo matrizes sejam possíveis, as matrizes consideradas devem ser conformáveis (mesma forma)
- Não confunda operações com matrizes em Fortran 90 com operações entre matrizes matemáticas

→ o produto $C = A \times B$ de duas matrizes, $A_{mn}B_{nl}$, produzirá uma matriz C_{ml} numa multiplicação entre matrizes matemáticas

Operações com Matrizes

- Atribuir valor constante para uma Matriz

```
REAL, DIMENSION(20) :: A = 1.0, B = -1.0
INTEGER, DIMENSION(100) :: C
REAL, DIMENSION(:,:), ALLOCATABLE :: D
INTEGER :: N, M
READ*,N, M
ALLOCATE (D(N,M))
C = 5
D = 0.0
END
```

→ Todos os elementos da matriz são igualados à constante

→ Zere sempre uma matriz alocada

Operações com Matrizes

- Soma de Matrizes

$$A = \begin{pmatrix} 3 & 4 & 8 \\ 5 & 6 & 6 \end{pmatrix} \qquad B = \begin{pmatrix} 5 & 2 & 1 \\ 3 & 3 & 1 \end{pmatrix} \qquad C = A + B = \begin{pmatrix} 8 & 6 & 9 \\ 8 & 9 & 7 \end{pmatrix}$$

```
INTEGER, DIMENSION(2,3) :: A
INTEGER, DIMENSION(2,3) :: B
INTEGER, DIMENSION(2,3) :: C
A = reshape((/3,5,4,6,8,6/),shape=(/2,3/))
B = reshape((/5,3,2,3,1,1/),shape(B))
C = A + B
END
```

- A matriz resultante é produzida a partir da soma das matrizes, somando elemento a elemento

Operações com Matrizes

- Soma de um escalar por uma matriz

→ Um escalar (posto 0) é conformável com qualquer matriz

$$A = \begin{pmatrix} 3 & 4 & 8 \\ 5 & 6 & 6 \end{pmatrix} \quad 5 = \begin{pmatrix} 5 & 5 & 5 \\ 5 & 5 & 5 \end{pmatrix} \quad B = A + 5 = \begin{pmatrix} 8 & 9 & 13 \\ 10 & 11 & 11 \end{pmatrix}$$

```
INTEGER, DIMENSION(2,3) :: A
INTEGER, DIMENSION(2,3) :: B
A = reshape(source=(/3,5,4,6,8,6/),shape(/2,3/))
B = A + 5
print*,B
END
```

→ O escalar é distribuído em uma matriz, conformável com a matriz com que está sendo somado

Operações com Matrizes

- Multiplicação de Matrizes

$$A = \begin{pmatrix} 3 & 4 & 8 \\ 5 & 6 & 6 \end{pmatrix} \qquad B = \begin{pmatrix} 5 & 2 & 1 \\ 3 & 3 & 1 \end{pmatrix} \qquad C = A * B = \begin{pmatrix} 15 & 8 & 8 \\ 15 & 18 & 6 \end{pmatrix}$$

```
INTEGER, DIMENSION(2,3) :: A
INTEGER, DIMENSION(2,3) :: B
INTEGER, DIMENSION(2,3) :: C
A = reshape((/3,5,4,6,8,6/),shape=(/2,3/))
B = reshape((/5,3,2,3,1,1/),shape(B))
C = A * B
print*,C
END
```

- A matriz resultante é produzida a partir da multiplicação das matrizes, multiplicando elemento a elemento

Operações com Matrizes

- Divisão de Matrizes

$$A = \begin{pmatrix} 3 & 4 & 8 \\ 5 & 6 & 6 \end{pmatrix} \qquad B = \begin{pmatrix} 5 & 2 & 1 \\ 3 & 3 & 1 \end{pmatrix} \qquad C = A / B = \begin{pmatrix} 3/5 & 2 & 8 \\ 5/3 & 2 & 6 \end{pmatrix}$$

```
INTEGER, DIMENSION(2,3) :: A
INTEGER, DIMENSION(2,3) :: B
INTEGER, DIMENSION(2,3) :: C
A = reshape((/3,5,4,6,8,6/),shape=(/2,3/))
B = reshape((/5,3,2,3,1,1/),shape(B))
C = A / B
print*,C
END
```

- A matriz resultante é produzida a partir da divisão das matrizes, dividindo elemento a elemento

Funções Intrínsecas para Matrizes

- Maior dos elementos de uma matriz A: MAXVAL(A)
- Menor dos elementos de uma matriz A: MINVAL(A)

```
program max_min_valores
  implicit none
  integer, parameter :: n = 4
  integer, dimension(n) :: dados

  print*,"digite 4 inteiros quaisquer:"
  read*,dados(1),dados(2),dados(3),dados(4)
  print*,"dados --->",dados
  print*,"maior valor =",MAXVAL(dados)
  print*,"menor valor =",MINVAL(dados)

end program max_min_valores
```

Funções Intrínsecas para Matrizes

- Soma dos elementos de uma matriz A: SUM(A)
- Produto dos elementos de uma matriz A: PRODUCT(A)

```
program soma_produto
  implicit none
  integer, parameter :: n = 4
  integer, dimension(n) :: dados

  print*,"digite 4 inteiros quaisquer:"
  read*,dados(1),dados(2),dados(3),dados(4)
  print*,"dados -->",dados
  print*," soma dos elementos =",SUM(dados)
  print*," produto dos elementos =",PRODUCT(dados)

end program soma_produto
```

Funções Intrínsecas para Matrizes

- Transposta de uma matriz A de posto 2: TRANSPOSE(A)

```
program transposta_matriz
  implicit none
  real, dimension(3,2) :: A

  print*,"Digite os elementos reais de uma matriz A(3,2):"
  read*,A(1,1), A(2,1), A(3,1), A(1,2), A(2,2), A(3,2)
  print*,'Matriz A -> ',A
  print*,'Transposta da matriz A -> ', TRANSPOSE(A)

end program transposta_matriz
```

→ Se a matriz é da forma DIMENSION(M,N) a transposta tem forma DIMENSION(N,M)