# Matrix manipulation in Matlab and Mathematica

Symbol definitions:
`i, j, k, l, m, n`: integers.
`A`: $n \times m$ matrix. (e.g. Matlab: $A = [1, 2, 3; 4, 5, 6]$, Mathematica: $A = \{\{1, 2, 3\}, \{4, 5, 6\}\}$)
`b`: $n \times 1$ matrix (column vector). (e.g. Matlab: $b = [1; -2; 3]$, Mathematica: $b = \{1, -2, 3\}$)
`r`: $1 \times n$ matrix (row vector). (e.g. Matlab: $r = [1, 2, -3]$, Mathematica: $r = \{1, 2, -3\}$ (Mathematica automatically treat it as column or row vector, the one which is sensible))
`ids`: vector contains indexs. (e.g. Matlab: ids $= [3, 1, 2]$, Mathematica: ids $= \{3, 1, 2\}$)
`bools`: boolean vector. (e.g. Matlab: *bools* $= [$`false`, `true`, `true`$]$, Mathematica: *bools* $= \{$`False`, `True`, `True`$\}$)
Matlab use Column-major order. Mathematica use Row-major order.

| Matlab | Mathematica |
|---|---|
| `A(i, j)` | `A[[i, j]]` |
| `A(end-i+1, k)` | `A[[-i, k]]` |
| `A(i:j, k)` | `A[[i;;j, k]]` |
| `A(i:1:j, k)` | `A[[i;;j;;1, k]]` |
| `b(1:i)` | `b[[;;i]]` |
| `b(end-i+1:end)` | `b[[-i;;]]` |
| `A(:, k)` | `A[[;;, k]]` |
| `A(j, :)` | `A[[j]]` |
| `A(id, k)` | `A[[id, k]]` |
| `A(i)` | `??` |
| `A(:)` | `Flatten[Transpose[A]]` |

| Matlab | Mathematica |
|---|---|
| `b(bools)` | `Pick[b, bools]` |
| `find(b==1)` | `Position[b, 1]` |
| `b>0` | `#>0&/@b, Map[#>0&, b]` |
| `A>0` | `Map[#>0&, A, {2}]` |
| `1*(b>0)` | `1-UnitStep[-b]` |
| `b(b==1)` | `Select[b, #==1&]` |
| `r(b>0)` | `Pick[r, #>0&/@b]` |
| `fliplr(r)` | `Reverse[r]` |
| `A.'` | `Transpose[A]` |
| `permute(arr, ids)` | `Transpose[arr, ids]` |
| `sin(A)` | `Sin[A]` |

Table 1: Elementary operations

Note: In Mathematica, Indexing operation can be performed by `Part[]`. e.g. `A[[i, j]] == Part[A, i, j]`, special case is `b[[i;;j;;1]]==Part[b,i;;j;;1]==Take[b, {i,j,1}]`. ";;" is called `Span`.

| Matlab | (*index*) | `i` | `end-i+1` | `i:j` | `i:di:j` | `1:j` | `end-j+1:end` | `:` | `ids` | `bools` |
|---|---|---|---|---|---|---|---|---|---|---|
| Mathematica(Part[]) | [[*index*]] | `i` | `-i` | `i;;j` | `i;;j;;di` | `;;j` | `-j;;` | `;;` | `ids` | `??` |
| Mathematica(function) | | `{i}` | `{-i}` | `{i,j}` | `{i,j,di}` | `j` | `-j` | `All` | `??` | `??` |
| Python (numpy.array) | [*index*] | `i-1` | `-i` | `i-1:j` | `i-1:j:di` | `:j` | `-j:` | `:` | `ids` | `bools` |

Table 2: Summary: Indexing of vector (Indexing of one dimension of an array)

| MATLAB | MATHEMATICA |
|---|---|
| 1:n | Range[n] |
| m:k:n | Range[m, n, k] |
| zeros(m,n) | ConstantArray[0, {m,n}] |
| ones(m,n) | ConstantArray[1, {m,n}] |
| eye(n) | IdentityMatrix[n] |
| rand(m,n) | RandomReal[1, {m,n}] |
| f(1:n) | Table[f[i], {i,n}] |
| A(:) | Flatten[Transpose[A]] |
| reshape(v, d1, d2) | Partition[v, {d1,d2}] |

| MATLAB | MATHEMATICA |
|---|---|
| [v, x] | Append[v, x] |
| [x, v] | Prepend[v, x] |
| [v1, v2] | Join[v1, v2] |
| [v(1:k-1), x, v(k:end)] | Insert[v, x, k] |
| v(1:k) = [] | Drop[v, k] |
| v(j:k) = [] | Drop[v, {j, k}] |
| | |
| | |
| ['abc','def'] | ''abc''<>''def'' |

Table 3: Constructing/destructing a matrix

Note: strictly speaking, `reshape(v,d1,d2)` correspond to `Transpose[Partition[v, {d1,d2}]]`.

| MATLAB | MATHEMATICA |
|---|---|
| A+B | A+B |
| A*B | A.B |
| A.*B | A*B |
| A^n | MatrixPower[A,n] |
| A.^n | A^n |
| A' | ConjugateTranspose[A] |
| A.' | Transpose[A] |
| conj(A) | Conjugate[A] |
| | |
| b*r | KroneckerProduct[b,r] |
| r*b | r.b, b.r |
| diag(A) | Diagonal[A] |
| diag(b) | DiagonalMatrix[b] |
| | |

| MATLAB | MATHEMATICA |
|---|---|
| tr(A) | Tr(A) |
| det(A) | Det[A] |
| A \ b | LinearSolve[A,b] |
| ?? | LinearSolve[A] |
| inv(A) | Inverse(A) |
| eig(A) | Eigenvalues[A] |
| [vecs,vals]=eig(A) | {vals,vecs}=Eigensystem[A] |
| svd(A) | SingularValueList[A] |
| [s,v,d]=svd(A) | {s,v,d}=SingularValueDecomposition[A] |
| null(A) | NullSpace[A] |
| | |
| | |
| | |
| | |

Table 4: Common algebra operations

| Matlab | Mathematica |
|---|---|
| tic; toc; | AbsoluteTiming[] |
| | |

| Matlab | Mathematica |
|---|---|
| arrayfun | Thread |
| | |

Table 5: Other useful functions

# 1 Common Operation Examples

**Save matrix $A$ to plain text file "abc.txt", with full precision, in tab-delimited format**

Matlab

```
save('abc.txt', 'A', '-ascii', '-double'. '-tabs');
```

Mathematica

```
Export["abc.txt", A, "Table"];
```

Note: you may use "<>" to concatenate path, e.g.

```
NotebookDirectory[] <> "abc.txt"
```

**Load matrix $A$ from plain text file "abc.txt"**

Matlab

```
A=load('abc.txt');
```

Mathematica

```
A=Import["abc.txt", "Table"];
```

**Generate a meshgrided table**

Matlab

```
[ii,jj]=meshgrid(1:n,1:m);A=f(ii,jj)
```

Mathematica

```
A=Table[f[i,j],{i,n},{j,m}]
```

**Get high precision numbers**

Double to high precision number

```
In[1]:= SetPrecision[1.0/3, 40]
Out[1]= 0.3333333333333333314829616256247390992939 5
```

Exact number to high precision number

```
In[1]:= SetPrecision[1/3, 40] == N[1/3, 40]
Out[1]= True
```

**Number in specified base (16), with precision (20)**

```
In[1]:= 16^^FF.F‘20
Out[1]= 255.937500000000000000
In[2]:= 16^^F.FF‘20*^1
Out[2]= 255.937500000000000000
```

## 2 Ref

http://reference.wolfram.com/language/guide/ListManipulation.html
Input Syntax
http://reference.wolfram.com/language/tutorial/InputSyntax.html