# Built-in Operation (no library)

## Arithmetic Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| a=1 | a=1 | assignment |
| a+b | a+b | addition |
| a-b | a-b | subtraction |
| a\*b | a\*b | multiplication |
| a/b | a/b | division |
|  | a//b | floor division |
| a^b | a\*\*b pow(a,b) | power |
| rem(a,b) | a%b | remainder |

## Relational Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| a==b | a==b | equal |
| a<b | a<b | less than |
| a>b | a>b | greater than |
| a<=b | a<=b | less than or equal |
| a>=b | a>=b | greater than or equal |
| a~=b | a!=b | not equal |

## Logical Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| a&&b |  | short-circuit logical AND |
| a||b |  | short-circuit logical OR |
| a&b  and(a,b) | a and b | logical AND |
| a|b  or(a,b) | a or b | logical OR |
| xor(a,b) |  | logical EXCLUSIVE OR |
| ~a  not(a) | not a | logical NOT |
| any(a) |  | true if any element is nonzero |
| all(a) |  | true if all elements are nonzero |

## Complex Number Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| i,j,1i,1j | z=1j | imaginary unit |
| z=3+4i  z=3+4j | z=3+4j  z=complex(3,4) | a complex number |
| abs(z) | abs(z) | absolute value |
| real(z) | z.real | real part |
| imag(z) | z.imag | imaginary part |
| angle(z) |  | angle |
| conj(z) | z.conjugate() | complex conjugate |

## Set Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| unique(a) | set(a) | set unique |
| union(a,b) | a.union(b) | set union |
| intersect(a,b) | a.intersection(b) | set intersection |
| setdiff(a,b) | a.difference(b) | set difference |
| setxor(a,b) | a.symmetric\_difference(b) | set exclusion |
| ismember(elem,a) | elem in a | if an element is in an array / a set |

# Math Operation (math and numpy)

## Constants

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| pi | math.pi  np.pi | 3.141592 |
| exp(1) | math.e  np.e | 2.718281 |
| NaN  nan | math.nan  np.nan  np.NaN | not a number |
| Inf  inf | math.inf  np.inf  np.Inf | infinity |

## Functions

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| sqrt(a) | math.sqrt(a)  np.sqrt(a) | square root |
| log(a) | math.log(a)  np.log(a) | logarithm, base e |
| log10(a) | math.log10(a)  np.log10(a) | logarithm, base 10 |
| log2(a) | math.log2(a)  np.log2(a) | logarithm, base 2 |
| exp(a) | math.exp(a)  np.exp(a) | exponential function |
| factorial(a) | math.factorial(a)  np.math.factorial(a) | factorial |
| round(a) | round(a)  np.round(a) | round |
| ceil(a) | math.ceil(a)  np.ceil(a) | round up |
| floor(a) | math.floor(a)  np.floor(a) | round down |
| fix(a) | np.fix(a) | round towards zero |

# Random Number (random)

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| rand() | random.random() | uniform distribution between 0 and 1 |
|  | random.uniform(a,b) | uniform distribution between a and b |
| randn() | random.gauss(0,1) | standard normal distribution |

# Operating System (os)

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| dir  ls | os.listdir('.') | list files in current directory |
| pwd | os.getcwd() | displays current working directory |
| cd foo | os.chdir('foo') | change working directory |
| !notepad system("notepad") | os.system('notepad') os.popen('notepad') | invoke a system command |

# Array Operation (numpy)

In this section, variables for 1D arrays are denoted by lowercase letters, whereas variables for 2D arrays are denoted by uppercase letters.

## Create 1D Arrays

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| 1:10 | np.arange(1,11) | 1, 2, 3, ..., 10 |
| 1:3:10 | np.arange(1,11,3) | 1, 4, 7, 10 |
| 10:-1:1 | np.arange(10,0,-1) | 10, 9, 8, ..., 1 |
| 10:-3:1 | np.arange(10,0,-3) | 10, 7, 4, 1 |
| linspace(1,10,7) | np.linspace(1,10,7) | a linearly spaced vector from 1 to 10 (inclusive) with 7 points |

## Create 2D Arrays

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| [2,3;4,5] | np.array([[2,3],[4,5]]) | direct creation |
| zeros(3,5) | np.zeros((3,5)) | a 3x5 matrix with all zeros |
|  | np.empty((3,5)) | a 3x5 matrix without initialization |
| ones(3,5) | np.ones((3,5)) | a 3x5 matrix with all ones |
| eye(3) | np.identity(3) | a 3x3 identity matrix |
| diag([4,5,6]) | np.diag((4,5,6)) | a diagonal matrix |

## Assignment

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| A(:)=3 | A.fill(3)  A[:]=3 | set all values to the same scalar value |
| B=A | B=A.copy() | copy A to B |

## Indexing (Slicing)

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| a(2:end) | a[1:] | second to the last element |
| a(end) | a[-1] | the last element |
| a(end-1:end) | a[-2:] | last two elements |
| A(2,3) | A[1,2] | element at row 2 column 3 |
| A(1,:) | A[0]  A[0,]  A[0,:] | first row |
| A(:,1) | A[:,0]  A[...,0] | first column |
| A([1,3],[1,4]); | A[[0,2]][:,[0,3]] | use arrays as indices |
| A(2:end,:) | A[1:]  A[1:,]  A[1:,:] | all rows but the first row |
| A(end-1:end,:) | A[-2:]  A[-2:,]  A[-2:,:] | last two rows |
| A(1:2:end,:) | A[::2]  A[::2,]  A[::2,:] | every other row |

## Shape Query

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| size(A) | A.shape | array dimensions |
| length(A(:))  numel(A) | A.size  np.size(A) | number of elements |
| ndims(A) | A.ndim | number of dimensions  ndims of a MATLAB array is at least 2 |
| whos A | A.nbytes | number of bytes used in memory |

## Shape Changing

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
|  | np.concatenate((a,a)) | concatenate two vectors |
| [A;B] | np.concatenate((A,B))  np.concatenate((A,B),axis=0) np.vstack((A,B)) | concatenate along rows (1st axis) |
| [A,B] | np.concatenate((A,B),axis=1) np.hstack((A,B)) | concatenate along columns (2nd axis) |
|  | np.concatenate((A,B),axis=2) np.dstack((A,B)) | concatenate along depth (3rd axis) |
| [A(:);B(:)] | np.concatenate((A,B),axis=None) | concatenate along row and flatten. |
| reshape(1:6,3,2) | np.arange(1,7).reshape(2,3) | reshape  MATLAB fill columns first  Python fill rows first |
| A(:) | A.reshape(-1)  A.ravel()  A.flatten() | flatten a matrix to a vector  MATLAB to a column vector  reshape and ravel does not return a copy |
| fliplr(A) | A[:,::-1]  np.fliplr(A)  np.flip(A,axis=1) | flip left-right |
| rot90(A) | np.rot90(A) | rotate counterclockwise 90 degrees |
| repmat(A,2,3) | np.kron(np.ones((2,3)),A) | repeat A to [A, A, A ; A, A, A] |
| repelem(a,N) | a.repeat(N) | repeat elements N times  a should be a vector |

## Multiplication

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| A.\*B | A\*B  np.multiply(A,B) | elementwise multiplication |
| A\*B | A@B  np.matmul(A,B)  np.dot(A,B) | matrix multiplication |
|  | np.inner(A,B) | A·BT |
|  | np.outer(A,B) | np.outer(A.ravel(), B.ravel()) |
| kron(A,B) | np.kron(A,B) | Kronecker product |
| a/B |  | a·B-1 |
| A\b | np.linalg.solve(A,b)  np.linalg.lstsq(A,b) | A-1·b |
| dot(u,v) | np.dot(u,v)  u@v | dot product |
| dot(A,B) |  | column-wise vector dot product |
| cross(A,B) |  | column-wise vector cross product |

## Find Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| find(A) | A.ravel().nonzero() | linear indices of non-zero elements |
| [i,j]=find(A) | i,j=A.nonzero() i,j=np.where(A) | indices of non-zero elements |
| [i,j,v]=find(A) | v=A.compress((A!=0).flat) v=np.extract(A!=0,A) | indices and values of non-zero elements |

## Linear Algebra Operations

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| A.'  transpose(A) | A.T  A.transpose()  np.transpose(A) | standard transpose |
| A' |  | conjugate transpose |
| diag(A,0) | A.diagonal(offset=0) | diagonal (offset to the right by 0) |
| trace(A) | A.trace(offset=0) | sum along diagonal |
| conj(A) | A.conj(A) | conjugate |
| det(A) | np.linalg.det(A) | determinant |
| inv(A) | np.linalg.inv(A) | inverse |
| pinv(A) | np.linalg.pinv(A) | pseudo-inverse |
| norm(A,’fro’) | np.linalg.norm(A) | Frobenius norm |
| norm(A) | np.linalg.norm(A,ord=2) | maximum singular value |
| rank(A) | np.linalg.matrix\_rank(A) | rank |
| eig(A) | np.linalg.eig(A)[0] | eigenvalues |
| [V,D]=eig(A) | D,V=np.linalg.eig(A) | eigenvectors and eigenvalues |
| [U,S,V]=svd(A) | U,S,VT=np.linalg.svd(A) | singular value decomposition |
| chol(A) | np.linalg.cholesky(A) | Cholesky factorization |
| triu(A) | np.triu(A) | upper triangular |
| tril(A) | np.tril(A) | lower triangular |

## Dimension Reduction Operation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| cumsum(A) | A.cumsum(axis=0) | Cumulative sum (for each column) |
| sum(A) | sum(A)  A.sum(axis=0)  np.sum(A,axis=0) | sum of each column |
| sum(A')' | A.sum(axis=1)  np.sum(A,axis=1) | sum of each row |
| sum(sum(A)) | A.sum()  np.sum(A) | sum of all elements |
| mean(A)  mean(A,1) | A.mean(axis=0) np.mean(A,axis=0) | average / mean along columns  keepdims in Python available |
| mean(A,2) | A.mean(axis=1)  np.mean(A,axis=1) | average / mean along rows  keepdims in Python available |
| mean(A,'all') | A.mean()  np.mean(A) | average / mean for all elements  keepdims in Python available |
| median(A)  median(A,1) | np.median(A,axis=0) | median along columns  keepdims in Python available |
| median(A,2) | np.median(A,axis=1) | median along rows  keepdims in Python available |
| median(A,'all') | np.median(A) | median for all elements  keepdims in Python available |
| std(A)  std(A,[],1) | A.std(axis=0)  np.std(A,axis=0) | standard deviation along columns  keepdims in Python available |
| std(A,[],2) | A.std(axis=1)  np.std(A,axis=1) | standard deviation along rows  keepdims in Python available |
| std(A,[],'all') | A.std()  np.std(A) | standard deviation for all elements keepdims in Python available |
| var(A)  var(A,[],1) | A.var(axis=0)  np.var(A,axis=0) | variance along columns  keepdims in Python available |
| var(A,[],2) | A.var(axis=1)  np.var(A,axis=1) | variance along rows  keepdims in Python available |
| var(A,[],'all') | A.var()  np.var(A) | variance for all elements  keepdims in Python available |
| max(A) | A.max(axis=0)  np.max(A,axis=0)  np.amax(A,axis=0) | max in each column |
| max(A')' | A.max(axis=1)  np.max(A,axis=1)  np.amax(A,axis=1) | max in each row |
| max(max(A)) | A.max()  np.max(A)  np.amax(A) | max in array |

## More on Maximum

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| [v,i]=max(a) | v,i=a.max(0),a.argmax(0) | v is value whereas i is index |
| max(A,B) | np.maximum(A,B) | elementwise max |

## Convolution and Correlation

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| cov(X) | np.cov(X,rowvar=False) | covariance matrix between columns of X |
| cov(X,Y) |  | covariance matrix between flattened X and flattened Y |
| corr(X) | np.corrcoef(X,rowvar=False) | correlation coefficient matrix  element i, j is the correlation coefficient between column i in X and column j in X |
| corr(X,Y) |  | correlation coefficient matrix  element i, j is the correlation coefficient between column i in X and column j in Y |

## Sorting

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| sort(A(:)) | np.sort(A,axis=None) | flatten and sort |
| sort(A) | np.sort(A,axis=0)  np.msort(A) | sort each column |
| [~,I]=sort(A) | A.argsort(axis=0) | indices to sort each column |

## Difference and FFT

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| diff(A,N)  diff(A,N,1) | np.diff(A,n=N,axis=0) | difference between consecutive values applied N times for each column in A |
| diff(A,N,2) | np.diff(A,n=N)  np.diff(A,n=N,axis=1) | difference between consecutive values applied N times for each row in A |
| fft(A,N)  fft(A,N,1) | np.fft.fft(A,n=N,axis=0) | N point fast Fourier transform for each column in A, not divided by N |
| fft(A,N,2) | np.fft.fft(A,n=N)  np.fft.fft(A,n=N,axis=1) | N point fast Fourier transform for each row in A, not divided by N |
| ifft(A,N)  ifft(A,N,1) | np.fft.ifft(A,n=N,axis=0) | N point inverse Fourier transform for each column in A |
| ifft(A,N,2) | np.fft.ifft(A,n=N)  np.fft.ifft(A,n=N,axis=1) | N point inverse Fourier transform for each row in A |

## Set Operation (numpy)

a and b should be 1d arrays. If a and b are 2d arrays, they will be flattened.

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| unique(a) | np.unique(a) | set unique |
| union(a,b) | np.union1d(a,b) | set union |
| intersect(a,b) | np.intersect1d(a,b) | set intersection |
| setdiff(a,b) | np.setdiff1d(a,b) | set difference |
| setxor(a,b) | np.setxor1d(a,b) | set exclusion |
| ismember(elem,a) | elem in a | if an element is in an array / a set |

## Polynomials

|  |  |  |
| --- | --- | --- |
| MATLAB | Python | Description |
| p=polyfit(x,y,n) | p=np.polyfit(x,y,n) | fit polynomial with degree n to data  x and y should be vectors.  MATLAB allows for matrices whereas numpy does not. |
| polyval(p,x) | np.polyval(p,x) | evaluate polynomial p at x  x can be a vector or a matrix |
| roots(p) | np.roots(p) | find polynomial roots |