

NumPy for MATLAB users

Help

| MATLAB/Octave | Python | Description |
|--|----------------------------------|------------------------------------|
| <code>doc</code> | <code>help()</code> | Browse help interactively |
| <code>help -i % browse with Info</code> | | |
| <code>help help or doc doc</code> | <code>help</code> | Help on using help |
| <code>help plot</code> | <code>help(plot) or ?plot</code> | Help for a function |
| <code>help splines or doc splines</code> | <code>help(pylab)</code> | Help for a toolbox/library package |
| <code>demo</code> | | Demonstration examples |

Searching available documentation

| MATLAB/Octave | Python | Description |
|---------------------------|--|-------------------------|
| <code>lookfor plot</code> | | Search help files |
| <code>help</code> | <code>help(); modules [Numeric]</code> | List available packages |
| <code>which plot</code> | <code>help(plot)</code> | Locate functions |

Using interactively

| MATLAB/Octave | Python | Description |
|---------------------------------------|---|----------------------|
| <code>octave -q</code> | <code>ipython -pylab</code> | Start session |
| <code>TAB or M-?</code> | <code>TAB</code> | Auto completion |
| <code>foo(.m)</code> | <code>execfile('foo.py') or run foo.py</code> | Run code from file |
| <code>history</code> | <code>hist -n</code> | Command history |
| <code>diary on [...] diary off</code> | | Save command history |
| <code>exit or quit</code> | <code>CTRL-D</code> | End session |
| | <code>CTRL-Z # windows</code> | |
| | <code>sys.exit()</code> | |

Operators

| MATLAB/Octave | Python | Description |
|---------------------|--------|-------------------------|
| <code>help -</code> | | Help on operator syntax |

Arithmetic operators

| MATLAB/Octave | Python | Description |
|------------------------|-----------------------|-------------------------------|
| <code>a=1; b=2;</code> | <code>a=1; b=1</code> | Assignment; defining a number |

| | | |
|---------------------------|---|--|
| <code>a + b</code> | <code>a + b</code> <i>OR</i> <code>add(a,b)</code> | Addition |
| <code>a - b</code> | <code>a - b</code> <i>OR</i> <code>subtract(a,b)</code> | Subtraction |
| <code>a * b</code> | <code>a * b</code> <i>OR</i> <code>multiply(a,b)</code> | Multiplication |
| <code>a / b</code> | <code>a / b</code> <i>OR</i> <code>divide(a,b)</code> | Division |
| <code>a .^ b</code> | <code>a ** b</code> <code>power(a,b)</code> <code>pow(a,b)</code> | Power, a^b |
| <code>rem(a,b)</code> | <code>a % b</code> <code>remainder(a,b)</code> <code>fmod(a,b)</code> | Remainder |
| <code>a+=1</code> | <code>a+=b</code> <i>OR</i> <code>add(a,b,a)</code> | In place operation to save array creation overhead |
| <code>factorial(a)</code> | | Factorial, $n!$ |

Relational operators

| MATLAB/Octave | Python | Description |
|------------------------|--|-----------------------|
| <code>a == b</code> | <code>a == b</code> <i>OR</i> <code>equal(a,b)</code> | Equal |
| <code>a < b</code> | <code>a < b</code> <i>OR</i> <code>less(a,b)</code> | Less than |
| <code>a > b</code> | <code>a > b</code> <i>OR</i> <code>greater(a,b)</code> | Greater than |
| <code>a <= b</code> | <code>a <= b</code> <i>OR</i> <code>less_equal(a,b)</code> | Less than or equal |
| <code>a >= b</code> | <code>a >= b</code> <i>OR</i> <code>greater_equal(a,b)</code> | Greater than or equal |
| <code>a ~= b</code> | <code>a != b</code> <i>OR</i> <code>not_equal(a,b)</code> | Not Equal |

Logical operators

| MATLAB/Octave | Python | Description |
|--|--|----------------------------------|
| <code>a && b</code> | <code>a and b</code> | Short-circuit logical AND |
| <code>a b</code> | <code>a or b</code> | Short-circuit logical OR |
| <code>a & b</code> <i>OR</i> <code>and(a,b)</code> | <code>logical_and(a,b)</code> <i>OR</i> <code>a and b</code> | Element-wise logical AND |
| <code>a b</code> <i>OR</i> <code>or(a,b)</code> | <code>logical_or(a,b)</code> <i>OR</i> <code>a or b</code> | Element-wise logical OR |
| <code>xor(a, b)</code> | <code>logical_xor(a,b)</code> | Logical EXCLUSIVE OR |
| <code>~a</code> <i>OR</i> <code>not(a)</code> | <code>logical_not(a)</code> <i>OR</i> <code>not a</code> | Logical NOT |
| <code>~a</code> <i>OR</i> <code>!a</code> | | |
| <code>any(a)</code> | | True if any element is nonzero |
| <code>all(a)</code> | | True if all elements are nonzero |

root and logarithm

| MATLAB/Octave | Python | Description |
|-----------------------|----------------------------|-------------------------------|
| <code>sqrt(a)</code> | <code>math.sqrt(a)</code> | Square root |
| <code>log(a)</code> | <code>math.log(a)</code> | Logarithm, base e (natural) |
| <code>log10(a)</code> | <code>math.log10(a)</code> | Logarithm, base 10 |

| | | |
|----------------------|-----------------------------|----------------------------|
| <code>log2(a)</code> | <code>math.log(a, 2)</code> | Logarithm, base 2 (binary) |
| <code>exp(a)</code> | <code>math.exp(a)</code> | Exponential function |

Round off

| MATLAB/Octave | Python | Description |
|-----------------------|--|--------------------|
| <code>round(a)</code> | <code>round(a)</code> <i>or</i> <code>math.round(a)</code> | Round |
| <code>ceil(a)</code> | <code>ceil(a)</code> | Round up |
| <code>floor(a)</code> | <code>floor(a)</code> | Round down |
| <code>fix(a)</code> | <code>fix(a)</code> | Round towards zero |

Mathematical constants

| MATLAB/Octave | Python | Description |
|---------------------|--|----------------|
| <code>pi</code> | <code>math.pi</code> | $\pi=3.141592$ |
| <code>exp(1)</code> | <code>math.e</code> <i>or</i> <code>math.exp(1)</code> | $e=2.718281$ |

Missing values; IEEE-754 floating point status flags

| MATLAB/Octave | Python | Description |
|------------------|-------------------------|---------------------|
| <code>NaN</code> | <code>nan</code> | Not a Number |
| <code>Inf</code> | <code>inf</code> | Infinity, $+\infty$ |
| | <code>plus_inf</code> | Infinity, $+\infty$ |
| | <code>minus_inf</code> | Infinity, $-\infty$ |
| | <code>plus_zero</code> | Plus zero, $+0$ |
| | <code>minus_zero</code> | Minus zero, -0 |

Complex numbers

| MATLAB/Octave | Python | Description |
|-----------------------|---|--------------------------|
| <code>i</code> | <code>1j</code> | Imaginary unit |
| <code>z = 3+4i</code> | <code>z = 3+4j</code> <i>or</i> <code>z = complex(3,4)</code> | A complex number, $3+4i$ |
| <code>abs(z)</code> | <code>abs(3+4j)</code> | Absolute value (modulus) |
| <code>real(z)</code> | <code>z.real</code> | Real part |
| <code>imag(z)</code> | <code>z.imag</code> | Imaginary part |
| <code>arg(z)</code> | | Argument |
| <code>conj(z)</code> | <code>z.conj()</code> ; <code>z.conjugate()</code> | Complex conjugate |

Trigonometry

| MATLAB/Octave | Python | Description |
|------------------------|-------------------------|--------------------------------|
| <code>atan(a,b)</code> | <code>atan2(b,a)</code> | Arctangent, $\arctan(b/a)$ |
| | <code>hypot(x,y)</code> | Hypotenuse; Euclidean distance |

Generate random numbers

| MATLAB/Octave | Python | Description |
|-----------------------------|---|----------------------------------|
| <code>rand(1,10)</code> | <code>random.random((10,))</code> <code>random.uniform((10,))</code> | Uniform distribution |
| <code>2+5*rand(1,10)</code> | <code>random.uniform(2,7,(10,))</code> | Uniform: Numbers between 2 and 7 |
| <code>rand(6)</code> | <code>random.uniform(0,1,(6,6))</code> | Uniform: 6,6 array |
| <code>randn(1,10)</code> | <code>random.standard_normal((10,))</code> | Normal distribution |

Vectors

| MATLAB/Octave | Python | Description |
|--------------------------------|--|-------------------------------------|
| <code>a=[2 3 4 5];</code> | <code>a=array([2,3,4,5])</code> | Row vector, $1 \times n$ -matrix |
| <code>adash=[2 3 4 5]';</code> | <code>array([2,3,4,5])[:,NewAxis]</code> <code>array([2,3,4,5]).reshape(-1,1)</code> <code>r_[1:10,'c']</code> | Column vector, $m \times 1$ -matrix |

Sequences

| MATLAB/Octave | Python | Description |
|-------------------------------|--|--|
| <code>1:10</code> | <code>arange(1,11, dtype=Float)</code> <code>range(1,11)</code> | 1,2,3, ... ,10 |
| <code>0:9</code> | <code>arange(10.)</code> | 0.0,1.0,2.0, ... ,9.0 |
| <code>1:3:10</code> | <code>arange(1,11,3)</code> | 1,4,7,10 |
| <code>10:-1:1</code> | <code>arange(10,0,-1)</code> | 10,9,8, ... ,1 |
| <code>10:-3:1</code> | <code>arange(10,0,-3)</code> | 10,7,4,1 |
| <code>linspace(1,10,7)</code> | <code>linspace(1,10,7)</code> | Linearly spaced vector of $n=7$ points |
| <code>reverse(a)</code> | <code>a[::-1]</code> <i>or</i> | Reverse |
| <code>a(:) = 3</code> | <code>a.fill(3)</code> , <code>a[:] = 3</code> | Set all values to same scalar value |

Concatenation (vectors)

| MATLAB/Octave | Python | Description |
|----------------------|--|-------------------------|
| <code>[a a]</code> | <code>concatenate((a,a))</code> | Concatenate two vectors |
| <code>[1:4 a]</code> | <code>concatenate((range(1,5),a), axis=1)</code> | |

Repeating

| MATLAB/Octave | Python | Description |
|--------------------|---------------------------------|--------------|
| <code>[a a]</code> | <code>concatenate((a,a))</code> | 1 2 3, 1 2 3 |

| | |
|------------------------------------|----------------------------------|
| <code>a.repeat(3)</code> <i>or</i> | <code>1 1 1, 2 2 2, 3 3 3</code> |
| <code>a.repeat(a)</code> <i>or</i> | <code>1, 2 2, 3 3 3</code> |

Miss those elements out

| MATLAB/Octave | Python | Description |
|---------------------------|---------------------|------------------------|
| <code>a(2:end)</code> | <code>a[1:]</code> | miss the first element |
| <code>a([1:9])</code> | | miss the tenth element |
| <code>a(end)</code> | <code>a[-1]</code> | last element |
| <code>a(end-1:end)</code> | <code>a[-2:]</code> | last two elements |

Maximum and minimum

| MATLAB/Octave | Python | Description |
|-----------------------------|---|----------------------------------|
| <code>max(a,b)</code> | <code>maximum(a,b)</code> | pairwise max |
| <code>max([a b])</code> | <code>concatenate((a,b)).max()</code> | max of all values in two vectors |
| <code>[v,i] = max(a)</code> | <code>v,i = a.max(0),a.argmax(0)</code> | |

Vector multiplication

| MATLAB/Octave | Python | Description |
|-----------------------|-----------------------|---------------------------------|
| <code>a.*a</code> | <code>a*a</code> | Multiply two vectors |
| <code>dot(u,v)</code> | <code>dot(u,v)</code> | Vector dot product, $u \cdot v$ |

Matrices

| MATLAB/Octave | Python | Description |
|----------------------------|---------------------------------------|-----------------|
| <code>a = [2 3;4 5]</code> | <code>a = array([[2,3],[4,5]])</code> | Define a matrix |

Concatenation (matrices); rbind and cbind

| MATLAB/Octave | Python | Description |
|---------------------------|---|--------------------------------------|
| <code>[a ; b]</code> | <code>concatenate((a,b), axis=0)</code> <code>vstack((a,b))</code> | Bind rows |
| <code>[a , b]</code> | <code>concatenate((a,b), axis=1)</code> <code>hstack((a,b))</code> | Bind columns |
| | <code>concatenate((a,b), axis=2)</code> <code>dstack((a,b))</code> | Bind slices (three-way arrays) |
| <code>[a(:), b(:)]</code> | <code>concatenate((a,b), axis=None)</code> | Concatenate matrices into one vector |
| <code>[1:4 ; 1:4]</code> | <code>concatenate((r_[1:5],r_[1:5])).reshape(2,-1)</code> <code>vstack((r_[1:5],r_[1:5]))</code> | Bind rows (from vectors) |
| <code>[1:4 ; 1:4]'</code> | | Bind columns (from vectors) |

Array creation

| MATLAB/Octave | Python | Description |
|----------------------------|---------------------------------|----------------------------|
| <code>zeros(3,5)</code> | <code>zeros((3,5),Float)</code> | 0 filled array |
| | <code>zeros((3,5))</code> | 0 filled array of integers |
| <code>ones(3,5)</code> | <code>ones((3,5),Float)</code> | 1 filled array |
| <code>ones(3,5)*9</code> | | Any number filled array |
| <code>eye(3)</code> | <code>identity(3)</code> | Identity matrix |
| <code>diag([4 5 6])</code> | <code>diag((4,5,6))</code> | Diagonal |
| <code>magic(3)</code> | | Magic squares; Lo Shu |
| | <code>a = empty((3,3))</code> | Empty array |

Reshape and flatten matrices

| MATLAB/Octave | Python | Description |
|--|--|---|
| <code>reshape(1:6,3,2)'</code> | <code>arange(1,7).reshape(2,-1)</code> <code>a.setshape(2,3)</code> | Reshaping (rows first) |
| <code>reshape(1:6,2,3);</code> <code>a'(:)</code> | <code>arange(1,7).reshape(-1,2).transpose()</code> <code>a.flatten()</code> <i>or</i> | Reshaping (columns first) Flatten to vector (by rows, like comics) |
| <code>a(:)</code> | <code>a.flatten(1)</code> | Flatten to vector (by columns) |
| <code>vech(a)</code> | | Flatten upper triangle (by columns) |

Shared data (slicing)

| MATLAB/Octave | Python | Description |
|--------------------|---------------------------|-------------|
| <code>b = a</code> | <code>b = a.copy()</code> | Copy of a |

Indexing and accessing elements (Python: slicing)

| MATLAB/Octave | Python | Description |
|--|--|--------------------------|
| <code>a = [11 12 13 14 ... 21 22 23 24 ... 31 32 33 34]</code> | <code>a = array([[11, 12, 13, 14], [21, 22, 23, 24], [31, 32, 33, 34]])</code> | Input is a 3,4 array |
| <code>a(2,3)</code> | <code>a[1,2]</code> | Element 2,3 (row,col) |
| <code>a(1,:)</code> | <code>a[0,]</code> | First row |
| <code>a(:,1)</code> | <code>a[:,0]</code> | First column |
| <code>a([1 3],[1 4]);</code> | <code>a.take([0,2]).take([0,3], axis=1)</code> | Array as indices |
| <code>a(2:end,:)</code> | <code>a[1:,:]</code> | All, except first row |
| <code>a(end-1:end,:)</code> | <code>a[-2:,:]</code> | Last two rows |
| <code>a(1:2:end,:)</code> | <code>a[:,2:,:]</code> | Strides: Every other row |

| | | |
|----------------------------|--|--------------------------------|
| <code>a(:, [1 3 4])</code> | <code>a[... , 2]</code> | Third in last dimension (axis) |
| | <code>a.take([0, 2, 3], axis=1)</code> | Remove one column |
| | <code>a.diagonal(offset=0)</code> | Diagonal |

Assignment

| MATLAB/Octave | Python | Description |
|------------------------------------|--|--|
| <code>a(:, 1) = 99</code> | <code>a[:, 0] = 99</code> | |
| <code>a(:, 1) = [99 98 97]'</code> | <code>a[:, 0] = array([99, 98, 97])</code> | |
| <code>a(a > 90) = 90;</code> | <code>(a > 90).choose(a, 90)</code> | Clipping: Replace all elements over 90 |
| | <code>a.clip(min=None, max=90)</code> | |
| | <code>a.clip(min=2, max=5)</code> | Clip upper and lower values |

Transpose and inverse

| MATLAB/Octave | Python | Description |
|---|--------------------------------------|-------------------------|
| <code>a'</code> | <code>a.conj().transpose()</code> | Transpose |
| <code>a.' <i>or</i> transpose(a)</code> | <code>a.transpose()</code> | Non-conjugate transpose |
| <code>det(a)</code> | <code>linalg.det(a) <i>or</i></code> | Determinant |
| <code>inv(a)</code> | <code>linalg.inv(a) <i>or</i></code> | Inverse |
| <code>pinv(a)</code> | <code>linalg.pinv(a)</code> | Pseudo-inverse |
| <code>norm(a)</code> | <code>norm(a)</code> | Norms |
| <code>eig(a)</code> | <code>linalg.eig(a)[0]</code> | Eigenvalues |
| <code>svd(a)</code> | <code>linalg.svd(a)</code> | Singular values |
| <code>chol(a)</code> | <code>linalg.cholesky(a)</code> | Cholesky factorization |
| <code>[v, l] = eig(a)</code> | <code>linalg.eig(a)[1]</code> | Eigenvectors |
| <code>rank(a)</code> | <code>rank(a)</code> | Rank |

Sum

| MATLAB/Octave | Python | Description |
|--------------------------|--------------------------------|--------------------------|
| <code>sum(a)</code> | <code>a.sum(axis=0)</code> | Sum of each column |
| <code>sum(a')</code> | <code>a.sum(axis=1)</code> | Sum of each row |
| <code>sum(sum(a))</code> | <code>a.sum()</code> | Sum of all elements |
| | <code>a.trace(offset=0)</code> | Sum along diagonal |
| <code>cumsum(a)</code> | <code>a.cumsum(axis=0)</code> | Cumulative sum (columns) |

Sorting

| MATLAB/Octave | Python | Description |
|--|---|-----------------|
| <code>a = [4 3 2 ; 2 8 6 ; 1 4 7]</code> | <code>a = array([[4, 3, 2], [2, 8, 6], [1, 4, 7]])</code> | Example data |
| <code>sort(a(:))</code> | <code>a.ravel().sort() <i>or</i></code> | Flat and sorted |

| | | |
|----------------------------|---|----------------------------------|
| <code>sort(a)</code> | <code>a.sort(axis=0)</code> <i>or</i> <code>msort(a)</code> | Sort each column |
| <code>sort(a')</code> | <code>a.sort(axis=1)</code> | Sort each row |
| <code>sortrows(a,1)</code> | <code>a[a[:,0].argsort(),:]</code> | Sort rows (by first row) |
| | <code>a.ravel().argsort()</code> | Sort, return indices |
| | <code>a.argsort(axis=0)</code> | Sort each column, return indices |
| | <code>a.argsort(axis=1)</code> | Sort each row, return indices |

Maximum and minimum

| MATLAB/Octave | Python | Description |
|-----------------------------|--|--------------------|
| <code>max(a)</code> | <code>a.max(0)</code> <i>or</i> <code>amax(a [,axis=0])</code> | max in each column |
| <code>max(a')</code> | <code>a.max(1)</code> <i>or</i> <code>amax(a, axis=1)</code> | max in each row |
| <code>max(max(a))</code> | <code>a.max()</code> <i>or</i> | max in array |
| <code>[v i] = max(a)</code> | | return indices, i |
| <code>max(b,c)</code> | <code>maximum(b,c)</code> | pairwise max |
| <code>cummax(a)</code> | | |
| | <code>a.ptp(); a.ptp(0)</code> | max-to-min range |

Matrix manipulation

| MATLAB/Octave | Python | Description |
|--------------------------------|---|----------------------------------|
| <code>fliplr(a)</code> | <code>fliplr(a)</code> <i>or</i> <code>a[:,::-1]</code> | Flip left-right |
| <code>flipud(a)</code> | <code>flipud(a)</code> <i>or</i> <code>a[::-1,:]</code> | Flip up-down |
| <code>rot90(a)</code> | <code>rot90(a)</code> | Rotate 90 degrees |
| <code>repmat(a,2,3)</code> | <code>kron(ones((2,3)),a)</code> | Repeat matrix: [a a a ; a a a] |
| <code>kron(ones(2,3),a)</code> | | |
| <code>triu(a)</code> | <code>triu(a)</code> | Triangular, upper |
| <code>tril(a)</code> | <code>tril(a)</code> | Triangular, lower |

Equivalents to "size"

| MATLAB/Octave | Python | Description |
|---|---|--------------------------------|
| <code>size(a)</code> | <code>a.shape</code> <i>or</i> <code>a.getshape()</code> | Matrix dimensions |
| <code>size(a,2)</code> <i>or</i> <code>length(a)</code> | <code>a.shape[1]</code> <i>or</i> <code>size(a, axis=1)</code> | Number of columns |
| <code>length(a(:))</code> | <code>a.size</code> <i>or</i> <code>size(a[, axis=None])</code> | Number of elements |
| <code>ndims(a)</code> | <code>a.ndim</code> | Number of dimensions |
| | <code>a.nbytes</code> | Number of bytes used in memory |

Matrix- and elementwise- multiplication

| MATLAB/Octave | Python | Description |
|---------------------|---|------------------------|
| <code>a .* b</code> | <code>a * b</code> <i>or</i> <code>multiply(a,b)</code> | Elementwise operations |

| | | |
|------------------------|--|--|
| <code>a * b</code> | <code>matrixmultiply(a,b)</code> <code>inner(a,b) <i>or</i></code> <code>outer(a,b) <i>or</i></code> | Matrix product (dot product) Inner matrix vector multiplication $a \cdot b'$ Outer product |
| <code>kron(a,b)</code> | <code>kron(a,b)</code> | Kronecker product |
| <code>a / b</code> | | Matrix division, $b \cdot a^{-1}$ |
| <code>a \ b</code> | <code>linalg.solve(a,b)</code> | Left matrix division, $b \cdot a^{-1}$ (solve linear equations) |
| | <code>vdot(a,b)</code> | Vector dot product |
| | <code>cross(a,b)</code> | Cross product |

Find; conditional indexing

| MATLAB/Octave | Python | Description |
|--------------------------------|--|----------------------------------|
| <code>find(a)</code> | <code>a.ravel().nonzero()</code> | Non-zero elements, indices |
| <code>[i j] = find(a)</code> | <code>(i,j) = a.nonzero()</code> <code>(i,j) = where(a!=0)</code> | Non-zero elements, array indices |
| <code>[i j v] = find(a)</code> | <code>v = a.compress((a!=0).flat)</code> <code>v = extract(a!=0,a)</code> | Vector of non-zero values |
| <code>find(a>5.5)</code> | <code>(a>5.5).nonzero()</code> | Condition, indices |
| | <code>a.compress((a>5.5).flat)</code> | Return values |
| <code>a .* (a>5.5)</code> | <code>where(a>5.5,0,a) <i>or</i> a * (a>5.5)</code> | Zero out elements above 5.5 |
| | <code>a.put(2,indices)</code> | Replace values |

Multi-way arrays

| MATLAB/Octave | Python | Description |
|---|--|----------------------|
| <code>a = cat(3, [1 2; 1 2],[3 4; 3 4]);</code> <code>a(1,:,:) </code> | <code>a = array([[[1,2],[1,2]], [[3,4],[3,4]])</code> <code>a[0,...]</code> | Define a 3-way array |

File input and output

| MATLAB/Octave | Python | Description |
|---|--|------------------------------|
| <code>f = load('data.txt')</code> | <code>f = fromfile("data.txt")</code> <code>f = load("data.txt")</code> | Reading from a file (2d) |
| <code>f = load('data.txt')</code> | <code>f = load("data.txt")</code> | Reading from a file (2d) |
| <code>x = dlmread('data.csv', ';')</code> | <code>f = load('data.csv', delimiter=';')</code> | Reading from a CSV file (2d) |
| <code>save -ascii data.txt f</code> | <code>save('data.csv', f, fmt='%.6f', delimiter=';')</code> | Writing to a file (2d) |

| | |
|--|--------------------------|
| <code>f.tofile(file='data.csv', format='%.6f', sep=';')</code> | Writing to a file (1d) |
| <code>f = fromfile(file='data.csv', sep=';')</code> | Reading from a file (1d) |

Plotting

Basic x-y plots

| MATLAB/Octave | Python | Description |
|--------------------------------------|---|--|
| <code>plot(a)</code> | <code>plot(a)</code> | 1d line plot |
| <code>plot(x(:,1),x(:,2),'o')</code> | <code>plot(x[:,0],x[:,1],'o')</code> | 2d scatter plot |
| <code>plot(x1,y1, x2,y2)</code> | <code>plot(x1,y1,'bo', x2,y2,'go')</code> | Two graphs in one plot |
| <code>plot(x1,y1)</code> | <code>plot(x1,y1,'o')</code> | Overplotting: Add new plots to current |
| <code>hold on</code> | <code>plot(x2,y2,'o')</code> | |
| <code>plot(x2,y2)</code> | <code>show() # as normal</code> | |
| <code>subplot(211)</code> | <code>subplot(211)</code> | subplots |
| <code>plot(x,y,'ro-')</code> | <code>plot(x,y,'ro-')</code> | Plotting symbols and color |

Axes and titles

| MATLAB/Octave | Python | Description |
|---------------------------------|------------------------------------|------------------------|
| <code>grid on</code> | <code>grid()</code> | Turn on grid lines |
| <code>axis equal</code> | <code>figure(figsize=(6,6))</code> | 1:1 aspect ratio |
| <code>axis('equal')</code> | | |
| <code>replot</code> | | |
| <code>axis([0 10 0 5])</code> | <code>axis([0, 10, 0, 5])</code> | Set axes manually |
| <code>title('title')</code> | | Axis labels and titles |
| <code>xlabel('x-axis')</code> | | |
| <code>ylabel('y-axis')</code> | | |
| | <code>text(2,25,'hello')</code> | Insert text |

Log plots

| MATLAB/Octave | Python | Description |
|--------------------------|--------------------------|--------------------------|
| <code>semilogy(a)</code> | <code>semilogy(a)</code> | logarithmic y-axis |
| <code>semilogx(a)</code> | <code>semilogx(a)</code> | logarithmic x-axis |
| <code>loglog(a)</code> | <code>loglog(a)</code> | logarithmic x and y axes |

Filled plots and bar plots

| MATLAB/Octave | Python | Description |
|-------------------------------------|--|-------------|
| <code>fill(t,s,'b', t,c,'g')</code> | <code>fill(t,s,'b', t,c,'g', alpha=0.2)</code> | Filled plot |
| <code>% fill has a bug?</code> | | |

Functions

| MATLAB/Octave | Python | Description |
|---|--------------------------------------|---------------------------------|
| <code>f = inline('sin(x/3) - cos(x/5)')</code> | | Defining functions |
| <code>ezplot(f,[0,40])</code> | <code>x = arange(0,40,.5)</code> | Plot a function for given range |
| <code>fplot('sin(x/3) - cos(x/5)', [0,40])</code> | <code>y = sin(x/3) - cos(x/5)</code> | |
| <code>% no ezplot</code> | <code>plot(x,y, 'o')</code> | |

Polar plots

| MATLAB/Octave | Python | Description |
|-----------------------------------|---|-------------|
| <code>theta = 0:.001:2*pi;</code> | <code>theta = arange(0,2*pi,0.001)</code> | |
| <code>r = sin(2*theta);</code> | <code>r = sin(2*theta)</code> | |
| <code>polar(theta, rho)</code> | <code>polar(theta, rho)</code> | |

Histogram plots

| MATLAB/Octave | Python | Description |
|--|--------|-------------|
| <code>hist(randn(1000,1))</code> | | |
| <code>hist(randn(1000,1), -4:4)</code> | | |
| <code>plot(sort(a))</code> | | |

3d data

Contour and image plots

| MATLAB/Octave | Python | Description |
|--|---|-------------------------|
| <code>contour(z)</code> | <code>levels, colls = contour(Z, V, origin='lower', extent=(-3,3,-3,3))</code> | Contour plot |
| <code>contourf(z); colormap(gray)</code> | <code>clabel(colls, levels, inline=1, fmt='%1.1f', fontsize=10)</code> | Filled contour plot |
| | <code>contourf(Z, V, cmap=cm.gray, origin='lower', extent=(-3,3,-3,3))</code> | |
| <code>image(z)</code> | <code>im = imshow(Z, interpolation='bilinear', origin='lower', extent=(-3,3,-3,3))</code> | Plot image data |
| <code>colormap(gray)</code> | <code># imshow() and contour() as above</code> | Image with contours |
| <code>quiver()</code> | <code>quiver()</code> | Direction field vectors |

Perspective plots of surfaces over the x-y plane

| MATLAB/Octave | Python | Description |
|--|---|--------------------------------------|
| <pre>n=-2:.1:2; [x,y] = meshgrid(n,n); z=x.*exp(-x.^2-y.^2); mesh(z) surf(x,y,z) <i>or</i> surf1(x,y,z) % no surf1()</pre> | <pre>n=arrayrange(-2,2,.1) [x,y] = meshgrid(n,n) z = x*power(math.e,-x**2-y**2)</pre> | <p>Mesh plot</p> <p>Surface plot</p> |

Scatter (cloud) plots

| MATLAB/Octave | Python | Description |
|------------------------------|--------|-----------------|
| <pre>plot3(x,y,z,'k+')</pre> | | 3d scatter plot |

Save plot to a graphics file

| MATLAB/Octave | Python | Description |
|---|-------------------------------|-------------------------------|
| <pre>plot(1:10) print -depsc2 foo.eps gset output "foo.eps" gset terminal postscript eps plot(1:10)</pre> | <pre>savefig('foo.eps')</pre> | PostScript |
| | <pre>savefig('foo.pdf')</pre> | PDF |
| | <pre>savefig('foo.svg')</pre> | SVG (vector graphics for www) |
| <pre>print -dpng foo.png</pre> | <pre>savefig('foo.png')</pre> | PNG (raster graphics) |

Data analysis

Set membership operators

| MATLAB/Octave | Python | Description |
|--|--|------------------|
| <pre>a = [1 2 2 5 2]; b = [2 3 4];</pre> | <pre>a = array([1,2,2,5,2]) b = array([2,3,4]) a = set([1,2,2,5,2]) b = set([2,3,4])</pre> | Create sets |
| <pre>unique(a)</pre> | <pre>unique1d(a) unique(a) set(a)</pre> | Set unique |
| <pre>union(a,b)</pre> | <pre>union1d(a,b) a.union(b)</pre> | Set union |
| <pre>intersect(a,b)</pre> | <pre>intersect1d(a) a.intersection(b)</pre> | Set intersection |
| <pre>setdiff(a,b)</pre> | <pre>setdiff1d(a,b) a.difference(b)</pre> | Set difference |

| | | |
|----------------------------|--|---------------------|
| <code>setxor(a,b)</code> | <code>setxor1d(a,b)</code> | Set exclusion |
| <code>ismember(2,a)</code> | <code>a.symmetric_difference(b)</code> <code>2 in a</code> <code>setmember1d(2,a)</code> <code>contains(a,2)</code> | True for set member |

Statistics

| MATLAB/Octave | Python | Description |
|------------------------|--|-------------------------|
| <code>mean(a)</code> | <code>a.mean(axis=0)</code> <code>mean(a [,axis=0])</code> | Average |
| <code>median(a)</code> | <code>median(a)</code> <i>or</i> <code>median(a [,axis=0])</code> | Median |
| <code>std(a)</code> | <code>a.std(axis=0)</code> <i>or</i> <code>std(a [,axis=0])</code> | Standard deviation |
| <code>var(a)</code> | <code>a.var(axis=0)</code> <i>or</i> <code>var(a)</code> | Variance |
| <code>corr(x,y)</code> | <code>correlate(x,y)</code> <i>or</i> <code>corrcoef(x,y)</code> | Correlation coefficient |
| <code>cov(x,y)</code> | <code>cov(x,y)</code> | Covariance |

Interpolation and regression

| MATLAB/Octave | Python | Description |
|--|--|--|
| <code>z = polyval(polyfit(x,y,1),x)</code> <code>plot(x,y,'o', x,z ,'-')</code> <code>a = x\y</code> | <code>(a,b) = polyfit(x,y,1)</code> <code>plot(x,y,'o', x,a*x+b,'-')</code> <code>linalg.lstsq(x,y)</code> | Straight line fit Linear least squares $y = ax + b$ |
| <code>polyfit(x,y,3)</code> | <code>polyfit(x,y,3)</code> | Polynomial fit |

Non-linear methods

Polynomials, root finding

| MATLAB/Octave | Python | Description |
|---|---|--------------------------|
| | <code>poly()</code> | Polynomial |
| <code>roots([1 -1 -1])</code> | <code>roots()</code> | Find zeros of polynomial |
| <code>f = inline('1/x - (x-1)')</code> <code>fzero(f,1)</code> | | Find a zero near $x = 1$ |
| <code>solve('1/x = x-1')</code> | | Solve symbolic equations |
| <code>polyval([1 2 1 2],1:10)</code> | <code>polyval(array([1,2,1,2]),arange(1,11))</code> | Evaluate polynomial |

Differential equations

| MATLAB/Octave | Python | Description |
|---------------|--------|-------------|
|---------------|--------|-------------|

| | | |
|----------------------|-----------------------------------|---|
| <code>diff(a)</code> | <code>diff(x, n=1, axis=0)</code> | Discrete difference function and approximate derivative Solve differential equations |
|----------------------|-----------------------------------|---|

Fourier analysis

| MATLAB/Octave | Python | Description |
|----------------------|--------------------------------|---------------------------|
| <code>fft(a)</code> | <code>fft(a)</code> <i>or</i> | Fast fourier transform |
| <code>ifft(a)</code> | <code>ifft(a)</code> <i>or</i> | Inverse fourier transform |
| | <code>convolve(x,y)</code> | Linear convolution |

Symbolic algebra; calculus

| MATLAB/Octave | Python | Description |
|-----------------------|--------|---------------|
| <code>factor()</code> | | Factorization |

Programming

| MATLAB/Octave | Python | Description |
|--|--|-------------------------------|
| <code>.m</code> | <code>.py</code> | Script file extension |
| <code>%</code> <i>% or #</i> | <code>#</code> | Comment symbol (rest of line) |
| <code>% must be in MATLABPATH</code> <i>% must be in LOADPATH</i> | <code>from pylab import *</code> | Import library functions |
| <code>string='a=234';</code> <code>eval(string)</code> | <code>string="a=234"</code> <code>eval(string)</code> | Eval |

Loops

| MATLAB/Octave | Python | Description |
|--|---|--------------------------|
| <code>for i=1:5; disp(i); end</code> | <code>for i in range(1,6): print(i)</code> | for-statement |
| <code>for i=1:5</code> <code>disp(i)</code> <code>disp(i*2)</code> <code>end</code> | <code>for i in range(1,6):</code> <code>print(i)</code> <code>print(i*2)</code> | Multiline for statements |

Conditionals

| MATLAB/Octave | Python | Description |
|---|-------------------------------|-------------------|
| <code>if 1>0 a=100; end</code> | <code>if 1>0: a=100</code> | if-statement |
| <code>if 1>0 a=100; else a=0; end</code> | | if-else-statement |

Debugging

| MATLAB/Octave | Python | Description |
|---------------|--------|-------------|
|---------------|--------|-------------|

`ans`Most recent evaluated
expression`whos or who`List variables loaded into
memory`clear x or clear [all]`

Clear variable \$x\$ from memory

`disp(a)``print a`

Print

Working directory and OS

MATLAB/Octave

Python

Description

`dir or ls``os.listdir(".")`

List files in directory

`what``grep.grep("*.py")`

List script files in directory

`pwd``os.getcwd()`Displays the current working
directory`cd foo``os.chdir('foo')`

Change working directory

`!notepad``os.system('notepad')`

Invoke a System Command

`system("notepad")``os.popen('notepad')`

Time-stamp: "2007-11-09T16:46:36 vidar"

©2006 Vidar Bronken Gundersen, /mathesaurus.sf.net

Permission is granted to copy, distribute and/or modify this document as long as the above attribution is retained.