

MATLAB commands in numerical Python (NumPy)

Copyright ©2006 Vidar Bronken Gundersen

Permission is granted to copy, distribute and/or modify this document as long as the above attribution is kept and the resulting work is distributed under a license identical to this one.

The idea of this document (and the corresponding XML instance) is to provide a quick reference¹ for switching from MATLAB to an open-source environment, such as Python, Scilab, Octave and Gnuplot, or R for numeric processing and data visualisation.

Where Octave and Scilab commands are omitted, expect Matlab compatibility, and similarly where non given use the generic command.

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

1 Help

Desc.	MATLAB/Octave	Python	R
Browse help interactively	<code>doc</code>	<code>help()</code>	<code>help.start()</code>
Help on using help	<code>Octave:help -i % browse with Info</code>	<code>help</code>	<code>help()</code>
Help for a function	<code>help plot</code>	<code>help(plot)</code> or <code>?plot</code>	<code>help(plot)</code> or <code>?plot</code>
Help for a toolbox/library package	<code>help splines</code> or <code>doc splines</code>	<code>help(pylab)</code>	<code>help(package='splines')</code>
Demonstration examples	<code>demo</code>		<code>demo()</code>
Example using a function			<code>example(plot)</code>

1.1 Searching available documentation

Desc.	MATLAB/Octave	Python	R
Search help files	<code>lookfor plot</code>		<code>help.search('plot')</code>
Find objects by partial name			<code>apropos('plot')</code>
List available packages	<code>help</code>	<code>help(); modules [Numeric]</code>	<code>library()</code>
Locate functions	<code>which plot</code>	<code>help(plot)</code>	<code>find(plot)</code>
List available methods for a function			<code>methods(plot)</code>

1.2 Using interactively

Desc.	MATLAB/Octave	Python	R
Start session	<code>Octave:octave -q</code>	<code>ipython -pylab</code>	<code>Rgui</code>
Auto completion	<code>Octave:TAB</code> or <code>M-?</code>	<code>TAB</code>	
Run code from file	<code>foo(.m)</code>	<code>execfile('foo.py')</code> or <code>run foo.py</code>	<code>source('foo.R')</code>
Command history	<code>Octave:history</code>	<code>hist -n</code>	<code>history()</code>
Save command history	<code>diary on [..] diary off</code>		<code>savehistory(file=".Rhistory")</code>
End session	<code>exit</code> or <code>quit</code>	<code>CTRL-D</code> <code>CTRL-Z # windows</code> <code>sys.exit()</code>	<code>q(save='no')</code>

2 Operators

Desc.	MATLAB/Octave	Python	R
Help on operator syntax	<code>help -</code>		<code>help(Syntax)</code>

¹References: Hankin, Robin. *R for Octave users* (2001), available from <http://cran.r-project.org/doc/contrib/R-and-octave-2.txt> (accessed 2005.07.24); Martelli, Alex. *Python in a Nutshell* (O'Reilly, 2003); Oliphant, Travis. *Guide to NumPy* (Trelgol, 2006); Hunter, John. *The Matplotlib User's Guide* (2005), available from <http://matplotlib.sf.net/> (accessed 2005.07.31); Langtangen, Hans Petter. *Python Scripting for Computational Science* (Springer, 2004); Ascher et al.: *Numeric Python manual* (2001), available from <http://numeric.scipy.org/numpy.pdf> (accessed 2005.06.25); Moler, Cleve. *Numerical Computing with MATLAB* (MathWorks, 2004), available from <http://www.mathworks.com/moler/> (accessed 2005.03.10); Eaton, John W. *Octave Quick Reference* (1996); Merrit, Ethan. *Demo scripts for gnuplot version 4.0* (2004), available from <http://gnuplot.sourceforge.net/demo/> (accessed 2005.07.24); Woo, Alex. *Gnuplot Quick Reference* (2004), available from <http://www.gnuplot.info/docs/gpcard.pdf> (accessed 2005.07.14); Venables & Smith: *An Introduction to R* (2005), available from <http://cran.r-project.org/doc/manuals/R-intro.pdf> (accessed 2005.07.25); Short, Tom. *R reference card* (2005), available from <http://www.rpad.org/Rpad/R-refcard.pdf> (accessed 2005.07.24).

2.1 Arithmetic operators

Desc. Assignment; defining a number Addition Subtraction Multiplication Division Power, a^b	MATLAB/Octave <code>a=1; b=2;</code> <code>a + b</code> <code>a - b</code> <code>a * b</code> <code>a / b</code> <code>a .^ b</code>	Python <code>a=1; b=1</code> <code>a + b</code> or <code>add(a,b)</code> <code>a - b</code> or <code>subtract(a,b)</code> <code>a * b</code> or <code>multiply(a,b)</code> <code>a / b</code> or <code>divide(a,b)</code> <code>a ** b</code> <code>power(a,b)</code> <code>pow(a,b)</code> <code>a % b</code> <code>remainder(a,b)</code> <code>fmod(a,b)</code> <code>a+=b</code> or <code>add(a,b,a)</code>	R <code>a<-1; b<-2</code> <code>a + b</code> <code>a - b</code> <code>a * b</code> <code>a / b</code> <code>a ^ b</code> <code>a %% b</code> <code>a %/% b</code> <code>factorial(a)</code>
Remainder	<code>rem(a,b)</code>		
Integer division In place operation to save array creation overhead Factorial, $n!$	Octave: <code>a+=1</code> <code>factorial(a)</code>		

2.2 Relational operators

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

2.3 Logical operators

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

2.4 root and logarithm

Desc. Square root Logarithm, base e (natural) Logarithm, base 10 Logarithm, base 2 (binary) Exponential function	MATLAB/Octave <code>sqrt(a)</code> <code>log(a)</code> <code>log10(a)</code> <code>log2(a)</code> <code>exp(a)</code>	Python <code>math.sqrt(a)</code> <code>math.log(a)</code> <code>math.log10(a)</code> <code>math.log(a, 2)</code> <code>math.exp(a)</code>	R <code>sqrt(a)</code> <code>log(a)</code> <code>log10(a)</code> <code>log2(a)</code> <code>exp(a)</code>	\sqrt{a} $\ln a = \log_e a$ $\log_{10} a$ $\log_2 a$ e^a
---	--	--	--	--

2.5 Round off

Desc.	MATLAB/Octave	Python	R
Round	<code>round(a)</code>	<code>round(a)</code> or <code>math.round(a)</code>	<code>round(a)</code>
Round up	<code>ceil(a)</code>	<code>ceil(a)</code>	<code>ceil(a)</code>
Round down	<code>floor(a)</code>	<code>floor(a)</code>	<code>floor(a)</code>
Round towards zero	<code>fix(a)</code>	<code>fix(a)</code>	

2.6 Mathematical constants

Desc.	MATLAB/Octave	Python	R
$\pi = 3.141592$	<code>pi</code>	<code>math.pi</code>	<code>pi</code>
$e = 2.718281$	<code>exp(1)</code>	<code>math.e</code> or <code>math.exp(1)</code>	<code>exp(1)</code>

2.6.1 Missing values; IEEE-754 floating point status flags

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

2.7 Complex numbers

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

$$i = \sqrt{-1}$$

2.8 Trigonometry

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

$$\sqrt{x^2 + y^2}$$

2.9 Generate random numbers

Desc.	MATLAB/Octave	Python	R
Uniform distribution	<code>rand(1,10)</code>	<code>random.random((10,))</code> <code>random.uniform((10,))</code>	<code>runif(10)</code>
Uniform: Numbers between 2 and 7	<code>2+5*rand(1,10)</code>	<code>random.uniform(2,7,(10,))</code>	<code>runif(10, min=2, max=7)</code>
Uniform: 6,6 array	<code>rand(6)</code>	<code>random.uniform(0,1,(6,6))</code>	<code>matrix(runif(36),6)</code>
Normal distribution	<code>randn(1,10)</code>	<code>random.standard_normal((10,))</code>	<code>rnorm(10)</code>

3 Vectors

Desc. Row vector, $1 \times n$ -matrix Column vector, $m \times 1$ -matrix	MATLAB/Octave <code>a=[2 3 4 5];</code> <code>adash=[2 3 4 5]';</code>	Python <code>a=array([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>	R <code>a <- c(2,3,4,5)</code> <code>adash <- t(c(2,3,4,5))</code>
--	--	--	--

3.1 Sequences

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

3.2 Concatenation (vectors)

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

3.3 Repeating

Desc. <code>1 2 3, 1 2 3</code> <code>1 1 1, 2 2 2, 3 3 3</code> <code>1, 2 2, 3 3 3</code>	MATLAB/Octave <code>[a a]</code>	Python <code>concatenate((a,a))</code> <code>a.repeat(3) or</code> <code>a.repeat(a) or</code>	R <code>rep(a,times=2)</code> <code>rep(a,each=3)</code> <code>rep(a,a)</code>
--	-------------------------------------	---	---

3.4 Miss those elements out

Desc. miss the first element miss the tenth element miss 1,4,7, ... last element last two elements	MATLAB/Octave <code>a(2:end)</code> <code>a([1:9])</code> <code>a(end)</code> <code>a(end-1:end)</code>	Python <code>a[1:]</code> <code>a[-1]</code> <code>a[-2:]</code>	R <code>a[-1]</code> <code>a[-10]</code> <code>a[-seq(1,50,3)]</code>
---	---	---	--

3.5 Maximum and minimum

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

3.6 Vector multiplication

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

4 Matrices

Desc. Define a matrix	MATLAB/Octave <code>a = [2 3;4 5]</code>	Python <code>a = array([[2,3],[4,5]])</code>	R <code>rbind(c(2,3),c(4,5))</code> <code>array(c(2,3,4,5), dim=c(2,2))</code>
--------------------------	---	---	--

$$\begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$$

4.1 Concatenation (matrices); rbind and cbind

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

4.2 Array creation

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

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 9 & 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 & 9 \\ 9 & 9 & 9 & 9 & 9 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 4 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 6 \end{bmatrix}$$

$$\begin{bmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{bmatrix}$$

4.3 Reshape and flatten matrices

Desc.	MATLAB/Octave	Python	R	
Reshaping (rows first)	<code>reshape(1:6,3,2)'</code>	<code>arange(1,7).reshape(2,-1)</code> <code>a.setshape(2,3)</code>	<code>matrix(1:6,nrow=3,byrow=T)</code>	$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$
Reshaping (columns first)	<code>reshape(1:6,2,3)</code>	<code>arange(1,7).reshape(-1,2).transpose()</code>	<code>matrix(1:6,nrow=2)</code> <code>array(1:6,c(2,3))</code>	$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$
Flatten to vector (by rows, like comics)	<code>a'(:)</code>	<code>a.flatten()</code> or	<code>as.vector(t(a))</code>	$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 2 & 5 & 3 & 6 \end{bmatrix}$
Flatten to vector (by columns)	<code>a(:)</code>	<code>a.flatten(1)</code>	<code>as.vector(a)</code>	
Flatten upper triangle (by columns)	<code>vech(a)</code>		<code>a[row(a) <= col(a)]</code>	

4.4 Shared data (slicing)

Desc.	MATLAB/Octave	Python	R
Copy of a	<code>b = a</code>	<code>b = a.copy()</code>	<code>b = a</code>

4.5 Indexing and accessing elements (Python: slicing)

Desc.	MATLAB/Octave	Python	R	
Input is a 3,4 array	<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>	<code>a <- rbind(c(11, 12, 13, 14), c(21, 22, 23, 24), c(31, 32, 33, 34))</code>	$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$
Element 2,3 (row,col)	<code>a(2,3)</code>	<code>a[1,2]</code>	<code>a[2,3]</code>	a_{23}
First row	<code>a(1,:)</code>	<code>a[0,]</code>	<code>a[1,]</code>	$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$
First column	<code>a(:,1)</code>	<code>a[:,0]</code>	<code>a[,1]</code>	$\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}$
Array as indices	<code>a([1 3],[1 4]);</code>	<code>a.take([0,2]).take([0,3], axis=1)</code>		$\begin{bmatrix} a_{11} & a_{14} \\ a_{31} & a_{34} \end{bmatrix}$
All, except first row	<code>a(2:end,:)</code>	<code>a[1:,:]</code>	<code>a[-1,]</code>	$\begin{bmatrix} a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$
Last two rows	<code>a(end-1:end,:)</code>	<code>a[-2:,:]</code>		$\begin{bmatrix} a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$
Strides: Every other row	<code>a(1:2:end,:)</code>	<code>a[::2,:]</code>		$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}$
Third in last dimension (axis)		<code>a[... ,2]</code>		
All, except row,column (2,3)			<code>a[-2,-3]</code>	$\begin{bmatrix} a_{11} & a_{13} & a_{14} \\ a_{31} & a_{33} & a_{34} \end{bmatrix}$
Remove one column	<code>a(:,[1 3 4])</code>	<code>a.take([0,2,3],axis=1)</code>	<code>a[:, -2]</code>	$\begin{bmatrix} a_{11} & a_{13} & a_{14} \\ a_{21} & a_{23} & a_{24} \\ a_{31} & a_{33} & a_{34} \end{bmatrix}$
Diagonal		<code>a.diagonal(offset=0)</code>		$\begin{bmatrix} a_{11} & a_{22} & a_{33} & a_{44} \end{bmatrix}$

4.6 Assignment

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

4.7 Transpose and inverse

Desc.	MATLAB/Octave	Python	R
Transpose	<code>a'</code>	<code>a.conj().transpose()</code>	<code>t(a)</code>
Non-conjugate transpose	<code>a.' or transpose(a)</code>	<code>a.transpose()</code>	
Determinant	<code>det(a)</code>	<code>linalg.det(a)</code> or	<code>det(a)</code>
Inverse	<code>inv(a)</code>	<code>linalg.inv(a)</code> or	<code>solve(a)</code>
Pseudo-inverse	<code>pinv(a)</code>	<code>linalg.pinv(a)</code>	<code>ginv(a)</code>
Norms	<code>norm(a)</code>	<code>norm(a)</code>	
Eigenvalues	<code>eig(a)</code>	<code>linalg.eig(a)[0]</code>	<code>eigen(a)\$values</code>
Singular values	<code>svd(a)</code>	<code>linalg.svd(a)</code>	<code>svd(a)\$d</code>
Cholesky factorization	<code>chol(a)</code>	<code>linalg.cholesky(a)</code>	
Eigenvectors	<code>[v,1] = eig(a)</code>	<code>linalg.eig(a)[1]</code>	<code>eigen(a)\$vectors</code>
Rank	<code>rank(a)</code>	<code>rank(a)</code>	<code>rank(a)</code>

4.8 Sum

Desc.	MATLAB/Octave	Python	R
Sum of each column	<code>sum(a)</code>	<code>a.sum(axis=0)</code>	<code>apply(a,2,sum)</code>
Sum of each row	<code>sum(a')</code>	<code>a.sum(axis=1)</code>	<code>apply(a,1,sum)</code>
Sum of all elements	<code>sum(sum(a))</code>	<code>a.sum()</code>	<code>sum(a)</code>
Sum along diagonal		<code>a.trace(offset=0)</code>	
Cumulative sum (columns)	<code>cumsum(a)</code>	<code>a.cumsum(axis=0)</code>	<code>apply(a,2,cumsum)</code>

4.9 Sorting

Desc.	MATLAB/Octave	Python	R
Example data	<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>	
Flat and sorted	<code>sort(a(:))</code>	<code>a.ravel().sort()</code> or	<code>t(sort(a))</code>
Sort each column	<code>sort(a)</code>	<code>a.sort(axis=0)</code> or <code>msort(a)</code>	<code>apply(a,2,sort)</code>
Sort each row	<code>sort(a')'</code>	<code>a.sort(axis=1)</code>	<code>t(apply(a,1,sort))</code>
Sort rows (by first row)	<code>sortrows(a,1)</code>	<code>a[a[:,0].argsort(),:]</code>	
Sort, return indices		<code>a.ravel().argsort()</code>	<code>order(a)</code>
Sort each column, return indices		<code>a.argsort(axis=0)</code>	
Sort each row, return indices		<code>a.argsort(axis=1)</code>	

$$\begin{bmatrix} 4 & 3 & 2 \\ 2 & 8 & 6 \\ 1 & 4 & 7 \\ 1 & 2 & 2 \\ 3 & 4 & 4 \\ 6 & 7 & 8 \\ 1 & 3 & 2 \\ 2 & 4 & 6 \\ 4 & 8 & 7 \\ 2 & 3 & 4 \\ 2 & 6 & 8 \\ 1 & 4 & 7 \\ 1 & 4 & 7 \\ 2 & 8 & 6 \\ 4 & 3 & 2 \end{bmatrix}$$

4.10 Maximum and minimum

Desc.	MATLAB/Octave	Python	R
max in each column	<code>max(a)</code>	<code>a.max(0)</code> or <code>amax(a [,axis=0])</code>	<code>apply(a,2,max)</code>
max in each row	<code>max(a')</code>	<code>a.max(1)</code> or <code>amax(a, axis=1)</code>	<code>apply(a,1,max)</code>
max in array	<code>max(max(a))</code>	<code>a.max()</code> or	<code>max(a)</code>
return indices, i	<code>[v i] = max(a)</code>		<code>i <- apply(a,1,which.max)</code>
pairwise max	<code>max(b,c)</code>	<code>maximum(b,c)</code>	<code>pmax(b,c)</code>
max-to-min range	<code>cummax(a)</code>	<code>a.ptp(); a.ptp(0)</code>	<code>apply(a,2,cummax)</code>

4.11 Matrix manipulation

Desc.	MATLAB/Octave	Python	R
Flip left-right	<code>fliplr(a)</code>	<code>fliplr(a)</code> or <code>a[:,::-1]</code>	<code>a[,4:1]</code>
Flip up-down	<code>flipud(a)</code>	<code>flipud(a)</code> or <code>a[::-1,]</code>	<code>a[3:1,]</code>
Rotate 90 degrees	<code>rot90(a)</code>	<code>rot90(a)</code>	
Repeat matrix: [a a a ; a a a]	<code>repmat(a,2,3)</code> Octave: <code>kron(ones(2,3),a)</code>	<code>kron(ones((2,3)),a)</code>	<code>kronecker(matrix(1,2,3),a)</code>
Triangular, upper	<code>triu(a)</code>	<code>triu(a)</code>	<code>a[lower.tri(a)] <- 0</code>
Triangular, lower	<code>tril(a)</code>	<code>tril(a)</code>	<code>a[upper.tri(a)] <- 0</code>

4.12 Equivalent to "size"

Desc.	MATLAB/Octave	Python	R
Matrix dimensions	<code>size(a)</code>	<code>a.shape</code> or <code>a.getshape()</code>	<code>dim(a)</code>
Number of columns	<code>size(a,2)</code> or <code>length(a)</code>	<code>a.shape[1]</code> or <code>size(a, axis=1)</code>	<code>ncol(a)</code>
Number of elements	<code>length(a(:))</code>	<code>a.size</code> or <code>size(a[, axis=None])</code>	<code>prod(dim(a))</code>
Number of dimensions	<code>ndims(a)</code>	<code>a.ndim</code>	
Number of bytes used in memory		<code>a.nbytes</code>	<code>object.size(a)</code>

4.13 Matrix- and elementwise- multiplication

Desc.	MATLAB/Octave	Python	R
Elementwise operations	<code>a .* b</code>	<code>a * b</code> or <code>multiply(a,b)</code>	<code>a * b</code>
Matrix product (dot product)	<code>a * b</code>	<code>matrixmultiply(a,b)</code>	<code>a %*% b</code>
Inner matrix vector multiplication $a \cdot b'$		<code>inner(a,b)</code> or	
Outer product		<code>outer(a,b)</code> or	<code>outer(a,b)</code> or <code>a %o% b</code>
Cross product			<code>crossprod(a,b)</code> or <code>t(a) %*% b</code>
Kronecker product	<code>kron(a,b)</code>	<code>kron(a,b)</code>	<code>kronecker(a,b)</code>
Matrix division, $b \cdot a^{-1}$	<code>a / b</code>		
Left matrix division, $b^{-1} \cdot a$ (solve linear equations)	<code>a \ b</code>	<code>linalg.solve(a,b)</code>	<code>solve(a,b)</code>
Vector dot product		<code>vdot(a,b)</code>	
Cross product		<code>cross(a,b)</code>	

$$\begin{bmatrix} 1 & 5 \\ 9 & 16 \end{bmatrix} \begin{bmatrix} 7 & 10 \\ 15 & 22 \end{bmatrix} = \begin{bmatrix} 5 & 11 \\ 11 & 25 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 6 & 8 \\ 3 & 6 & 9 & 12 \\ 4 & 8 & 12 & 16 \end{bmatrix} \begin{bmatrix} 10 & 14 \\ 14 & 20 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 2 & 4 \\ 3 & 4 & 6 & 8 \\ 3 & 6 & 4 & 8 \\ 9 & 12 & 12 & 16 \end{bmatrix}$$

$$Ax = b$$

4.14 Find; conditional indexing

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

5 Multi-way arrays

Desc.	MATLAB/Octave	Python	R
Define a 3-way array	<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>	

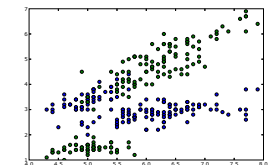
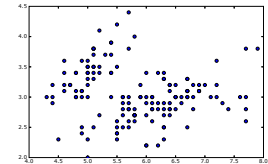
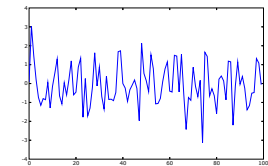
6 File input and output

Desc.	MATLAB/Octave	Python	R
Reading from a file (2d)	<code>f = load('data.txt')</code>	<code>f = fromfile("data.txt")</code> <code>f = load("data.txt")</code>	<code>f <- read.table("data.txt")</code>
Reading from a file (2d)	<code>f = load('data.txt')</code>	<code>f = load("data.txt")</code>	<code>f <- read.table("data.txt")</code>
Reading from a CSV file (2d)	<code>x = dlmread('data.csv', ',')</code>	<code>f = load('data.csv', delimiter=',')</code>	<code>f <- read.table(file="data.csv", sep=";")</code>
Writing to a file (2d)	<code>save -ascii data.txt f</code>	<code>save('data.csv', f, fmt='%.6f', delimiter=',', file="data.txt")</code>	<code>write(f, file="data.txt")</code>
Writing to a file (1d)		<code>f.tofile(file='data.csv', format='%.6f', sep=';')</code>	
Reading from a file (1d)		<code>f = fromfile(file='data.csv', sep=';')</code>	

7 Plotting

7.1 Basic x-y plots

Desc.	MATLAB/Octave	Python	R
1d line plot	<code>plot(a)</code>	<code>plot(a)</code>	<code>plot(a, type="l")</code>
2d scatter plot	<code>plot(x(:,1), x(:,2), 'o')</code>	<code>plot(x[:,0], x[:,1], 'o')</code>	<code>plot(x[,1], x[,2])</code>
Two graphs in one plot	<code>plot(x1,y1, x2,y2)</code>	<code>plot(x1,y1,'bo', x2,y2,'go')</code>	<code>plot(x1,y1)</code>
Overplotting: Add new plots to current	<code>plot(x1,y1)</code> <code>hold on</code> <code>plot(x2,y2)</code>	<code>plot(x1,y1,'o')</code> <code>plot(x2,y2,'o')</code> <code>show()</code> # as normal	<code>matplot(x2,y2,add=T)</code>
subplots	<code>subplot(211)</code>	<code>subplot(211)</code>	
Plotting symbols and color	<code>plot(x,y,'ro-')</code>	<code>plot(x,y,'ro-')</code>	<code>plot(x,y,type="b",col="red")</code>



7.1.1 Axes and titles

Desc.
Turn on grid lines
1:1 aspect ratio

MATLAB/Octave
`grid on`
`axis equal`
Octave:
`axis('equal')`
`replot`
`axis([0 10 0 5])`
`title('title')`
`xlabel('x-axis')`
`ylabel('y-axis')`

Python
`grid()`
`figure(figsize=(6,6))`

R
`grid()`
`plot(c(1:10,10:1), asp=1)`

Set axes manually
Axis labels and titles

`axis([0, 10, 0, 5])`

`plot(x,y, xlim=c(0,10), ylim=c(0,5))`
`plot(1:10, main="title",`
`xlab="x-axis", ylab="y-axis")`

Insert text

`text(2,25,'hello')`

7.1.2 Log plots

Desc.
logarithmic y-axis
logarithmic x-axis
logarithmic x and y axes

MATLAB/Octave
`semilogy(a)`
`semilogx(a)`
`loglog(a)`

Python
`semilogy(a)`
`semilogx(a)`
`loglog(a)`

R
`plot(x,y, log="y")`
`plot(x,y, log="x")`
`plot(x,y, log="xy")`

7.1.3 Filled plots and bar plots

Desc.

MATLAB/Octave

Python

R

Filled plot

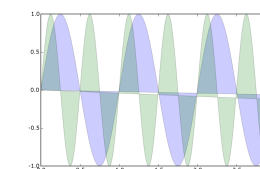
`fill(t,s,'b', t,c,'g')`
Octave: % fill has a bug?

`fill(t,s,'b', t,c,'g', alpha=0.2)`

`plot(t,s, type="n", xlab="", ylab="")`
`polygon(t,s, col="lightblue")`
`polygon(t,c, col="lightgreen")`

Stem-and-Leaf plot

`stem(x[,3])`



```
5 5
6 71
7 033
8 00113345567889
9 0133566677788
10 32674
```

7.1.4 Functions

Desc.
Defining functions

MATLAB/Octave
`f = inline('sin(x/3) - cos(x/5)')`

Python

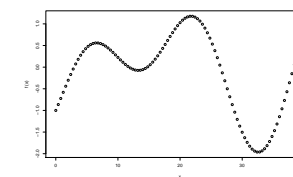
R
`f <- function(x) sin(x/3) - cos(x/5)` $f(x) = \sin\left(\frac{x}{3}\right) - \cos\left(\frac{x}{5}\right)$

Plot a function for given range

`ezplot(f,[0,40])`
`fplot('sin(x/3) - cos(x/5)',[0,40])`
Octave: % no ezplot

`x = arrayrange(0,40,.5)`
`y = sin(x/3) - cos(x/5)`
`plot(x,y, 'o')`

`plot(f, xlim=c(0,40), type='p')`



7.2 Polar plots

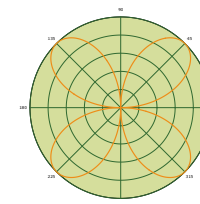
Desc.

```
MATLAB/Octave
theta = 0:.001:2*pi;
r = sin(2*theta);
```

```
Python
theta = arange(0,2*pi,0.001)
r = sin(2*theta)
```

R

$$\rho(\theta) = \sin(2\theta)$$



```
polar(theta, rho)
```

```
polar(theta, rho)
```

7.3 Histogram plots

Desc.

```
MATLAB/Octave
hist(randn(1000,1))
hist(randn(1000,1), -4:4)

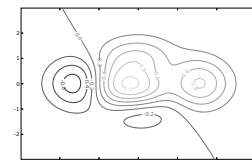
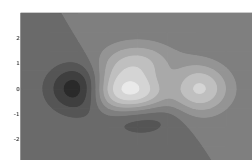

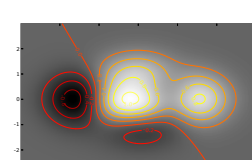
plot(sort(a))
```

Python

```
R
hist(rnorm(1000))
hist(rnorm(1000), breaks= -4:4)
hist(rnorm(1000), breaks=c(seq(-5,0,0.25), seq(0.5,5,0.5)), freq=F)
plot(apply(a,1,sort),type="l")
```

7.4 3d data

7.4.1 Contour and image plots

Desc.	MATLAB/Octave	Python	R	
Contour plot	<code>contour(z)</code>	<pre>levels, colls = contour(Z, V, origin='lower', extent=(-3,3,-3,3)) clabel(colls, levels, inline=1, fmt='%1.1f', fontsize=10)</pre>	<code>contour(z)</code>	
Filled contour plot	<code>contourf(z); colormap(gray)</code>	<pre>contourf(Z, V, cmap=cm.gray, origin='lower', extent=(-3,3,-3,3))</pre>	<code>filled.contour(x,y,z, nlevels=7, color=gray.colors)</code>	
Plot image data	<code>image(z)</code> <code>colormap(gray)</code>	<pre>im = imshow(Z, interpolation='bilinear', origin='lower', extent=(-3,3,-3,3))</pre>	<code>image(z, col=gray.colors(256))</code>	
Image with contours Direction field vectors	<code>quiver()</code>	<pre># imshow() and contour() as above quiver()</pre>		

7.4.2 Perspective plots of surfaces over the x-y plane

Desc.

MATLAB/Octave

```
n=-2:.1:2;
[x,y] = meshgrid(n,n);
z=x.*exp(-x.^2-y.^2);
```

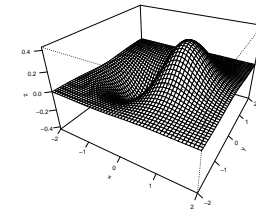
Python

```
n=arrayrange(-2,2,.1)
[x,y] = meshgrid(n,n)
z = x*power(math.e,-x**2-y**2)
```

R

```
f <- function(x,y) x*exp(-x^2-y^2)
n <- seq(-2,2, length=40)
z <- outer(n,n,f)
```

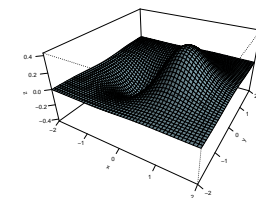
$$f(x,y) = xe^{-x^2-y^2}$$



Mesh plot

mesh(z)

```
persp(x,y,z,
      theta=30, phi=30, expand=0.6,
      ticktype='detailed')
```



Surface plot

surf(x,y,z) or surf1(x,y,z)
 Octave: % no surf1()

```
persp(x,y,z,
      theta=30, phi=30, expand=0.6,
      col='lightblue', shade=0.75, ltheta=120,
      ticktype='detailed')
```

7.4.3 Scatter (cloud) plots

Desc.

MATLAB/Octave

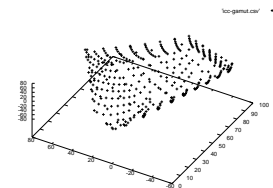
Python

R

3d scatter plot

plot3(x,y,z,'k')

cloud(z~x*y)



7.5 Save plot to a graphics file

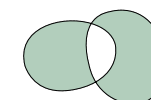
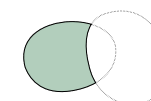
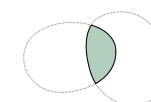
Desc. PostScript	MATLAB/Octave plot(1:10) print -depsc2 foo.eps Octave: gset output "foo.eps" gset terminal postscript eps plot(1:10)	Python savefig('foo.eps')	R postscript(file="foo.eps") plot(1:10) dev.off()
PDF SVG (vector graphics for www) PNG (raster graphics)	print -dpng foo.png	savefig('foo.pdf') savefig('foo.svg') savefig('foo.png')	pdf(file='foo.pdf') devSVG(file='foo.svg') png(filename = "Rplot%03d.png")

8 Data analysis

8.1 Set membership operators

Desc. Create sets	MATLAB/Octave a = [1 2 2 5 2]; b = [2 3 4];	Python a = array([1,2,2,5,2]) b = array([2,3,4]) a = set([1,2,2,5,2]) b = set([2,3,4])	R a <- c(1,2,2,5,2) b <- c(2,3,4)
Set unique	unique(a)	uniqueid(a) unique(a) set(a)	unique(a)
Set union	union(a,b)	union1d(a,b) a.union(b)	union(a,b)
Set intersection	intersect(a,b)	intersect1d(a) a.intersection(b)	intersect(a,b)
Set difference	setdiff(a,b)	setdiff1d(a,b) a.difference(b)	setdiff(a,b)
Set exclusion	setxor(a,b)	setxor1d(a,b) a.symmetric_difference(b)	setdiff(union(a,b),intersect(a,b))
True for set member	ismember(2,a)	2 in a setmember1d(2,a) contains(a,2)	is.element(2,a) or 2 %in% a

[1 2 5]



10 Programming

Desc. Script file extension Comment symbol (rest of line)	MATLAB/Octave .m % Octave: % or # % must be in MATLABPATH Octave: % must be in LOADPATH string='a=234'; eval(string)	Python .py # from pylab import * string="a=234" eval(string)	R .R # library(RSvgDevice) string <- "a <- 234" eval(parse(text=string))
Import library functions			
Eval			

10.1 Loops

Desc. for-statement Multiline for statements	MATLAB/Octave for i=1:5; disp(i); end for i=1:5 disp(i) disp(i*2) end	Python for i in range(1,6): print(i) for i in range(1,6): print(i) print(i*2)	R for(i in 1:5) print(i) for(i in 1:5) { print(i) print(i*2) }
--	--	---	---

10.2 Conditionals

Desc. if-statement if-else-statement Ternary operator (if?true:false)	MATLAB/Octave if 1>0 a=100; end if 1>0 a=100; else a=0; end	Python if 1>0: a=100	R if (1>0) a <- 100 ifelse(a>0,a,0) $a > 0 ? a : 0$
--	---	-------------------------	---

10.3 Debugging

Desc. Most recent evaluated expression List variables loaded into memory Clear variable x from memory Print	MATLAB/Octave ans whos or who clear x or clear [all] disp(a)	Python print a	R .Last.value objects() rm(x) print(a)
---	--	---------------------------	--

10.4 Working directory and OS

Desc. List files in directory List script files in directory Displays the current working directory Change working directory Invoke a System Command	MATLAB/Octave dir or ls what pwd cd foo !notepad Octave: system("notepad")	Python os.listdir(".") grep.grep("*.py") os.getcwd() os.chdir('foo') os.system('notepad') os.popen('notepad')	R list.files() or dir() list.files(pattern=".r\$") getwd() setwd('foo') system("notepad")
---	--	---	--

²This document is still draft quality. Most shown 2d plots are made using Matplotlib, and 3d plots using R and Gnuplot, provided as examples only.

³Version numbers and download URL for software used: Python 2.4.2, <http://www.python.org/>; NumPy 0.9.5, <http://numeric.scipy.org/>; Matplotlib 0.87, <http://matplotlib.sf.net/>; IPython 0.7.1, <http://ipython.scipy.org/>; R 2.1.1, <http://www.r-project.org/>; Octave 2.1.50, <http://www.octave.org/>; Scilab 4.0, <http://www.scilab.org/>; Gnuplot 4.0, <http://www.gnuplot.info/>.

⁴For referencing: Gundersen, Vidar Bronken. *MATLAB commands in numerical Python* (Oslo/Norway, 2005), available from: <http://mathesaurus.sf.net/>

⁵Contributions are appreciated: The best way to do this is to edit the XML and submit patches to our tracker or forums.