

Python Data Analysis Reference Card

Fabrizio Ferrari www.ferrari.pro.br

A,B: 2D array x,y: 1D vector M,N integers

Numpy Arrays

Creation	
<code>zeros((M,N))</code>	Returns array filled zeros, M lines, N columns
<code>ones((M,N))</code>	Returns array filled with ones, MxN
<code>empty((M,N))</code>	Returns array not filled (random values), MxN
<code>zeros_like(A)</code>	Return an array of zeros with shape and type of input.
<code>ones_like(A)</code>	Return an array of ones with shape and type of input.
<code>empty_like(A)</code>	Return an empty array with shape and type of input.
<code>random.random((M,N))</code>	Returns array filled with random numbers [0..1]
<code>identity(3,float)</code>	Identity 3x3 array of floats
<code>array([(1.5,2,3),(4,5,6)])</code>	specify values, 2x3 array of floats
<code>mgrid[1:3,2:5]</code>	rectangular mesh grid with x values [1,2] and y values [2,3,4]
<code>fromfunction(f, (3,3))</code>	Returns 3x3 array with function $f(i,j)$ evaluated for all combinations of indices
<code>arange(1., 10., 0.5)</code>	Array with range and step of values
<code>linspace(0,2,9)</code>	9 numbers from 0 to 2
Methods	
<code>A.sum()</code>	Sum array (may specify axis)
<code>A.min()</code>	Minimum value
<code>A.max()</code>	Maximum Value
<code>A.mean()</code>	Average value
<code>A.std()</code>	Standard Deviation
<code>A.var()</code>	Variance
<code>A.trace()</code>	Array trace
<code>A.size()</code>	Number of elements
<code>A.shape()</code>	Shape
<code>A.ptp()</code>	Peak-to-peak (maximum - minimum)
<code>A.ravel()</code>	1-d version of A
<code>A.transpose(), A.T</code>	transpose indices of array
<code>A.resize(M,N)</code>	Replicate or truncate array to new shape <i>in place</i>
<code>A.reshape(M,N)</code>	Returns array with new shape
<code>A.clip(Amin, Amax)</code>	Clip values of array a at values Amin, Amax
<code>A.compress(condition, axis=None)</code>	Selects elements from array A based on condition
<code>A.conjugate()</code>	Return an array with all complex-valued elements conjugated.
<code>A.copy</code>	Return a copy of the array.
<code>A.cumprod(axis=0)</code>	Cumulative product along specied axis.
<code>A.cumsum(axis=0)</code>	Return the cumulative sum of the elements along the given axis.
<code>A.diagonal(offset=0, axis1=0, axis2=1)</code>	Returns diagonal of 2D matrix with optional offsets.
<code>A.fill(value)</code>	fills in with a specified value

Numpy Arrays (cont.)

Operations	
<code>C = A-B</code>	<i>Arithmetic operations are elementwise</i>
<code>A**2</code>	Elementwise subtraction ($C_{i,j} = A_{i,j} - B_{i,j}$)
<code>dot(A,B)</code>	Returns array with each A element squared
<code>mat(A) * mat(B)</code>	Matrix product
<code>matrixmultiply(A, B)</code>	Matrix product
<code>inner(A, B)</code>	Matrix product
<code>outer(A, B)</code>	Inner product
<code>product(A, axis=0)</code>	Outer product
<code>concatenate(arrays, axis=0)</code>	Net product of elements along specified axis.
<code>vstack(A,B)</code>	Concatenate arrays contained in sequence of arrays
<code>hstack(A,B)</code>	Vertically stacks two arrays
<code>vsplit(A,2)</code>	Horizontally stacks two arrays
<code>hsplit(A,2)</code>	Vertically splits A in 2
	Horizontally Splits A in 2
Indexing	
<code>x[2]</code>	3rd elements (zero based indexing)
<code>x[-2]</code>	Counting from the end
<code>x[2:5]</code>	subarray, [<code>x[2]</code> , <code>x[3]</code> , <code>x[4]</code>]
<code>x[:5]</code>	from the beginning to the 4th element
<code>x[2:]</code>	from the 3rd to the end
<code>x[:]</code>	the whole array
<code>x[2:9:3]</code>	every 3 elements, [<code>x[2]</code> , <code>x[5]</code> , <code>x[8]</code>]
<code>x[numpy.where(x>7)]</code>	returns elements in x that satisfy criteria
<code>indices(shape, type=None)</code>	Generate array with values corresponding to position of selected index of the array
<code>A[j][i]</code>	Indexing convention (j row, i column)
<code>A[3][2]</code>	3rd element in the 4th row
<code>A[1]</code>	2nd row

Numpy Arrays Caution: The cumulative class of operations that either sum or multiply elements of an array (sum, product, cumsum, cumproduct) use the input array type as the type to perform the cumulative sum or product. If you are not using the largest type for the array (i.e., int64, Float64, etc.) you are asking for trouble. One is likely to run into integer overflows or Floating point precision problems if you don't specifically ask for a higher precision result, especially for large array. One should get into the habit of specifying the keyword argument dtype to a highest precision type when using these functions, e.g. `sum(arr, dtype=Float64)`.

Numpy Arrays (cont.)

<p>Questions</p> <p><code>all(A, axis=None)</code></p> <p><code>allclose(A, B, rtol=1.e-5, atol=1.e-8)</code></p> <p><code>alltrue(A, axis=0)</code> <code>any(A , axis=None)</code></p> <p><code>sometrue(A, axis=0)</code> <code>where(a)</code></p> <p>Ordering</p> <p><code>argmax(A, axis=-1), argmin(a,axis=-1)</code></p> <p><code>argsort(A, axis=-1)</code> <code>searchsorted(bin, A)</code></p> <p><code>sort(A, axis=-1)</code> <code>choose(selector, population, clipmode=CLIP)</code></p> <p>File Ops</p> <p><code>fromfile(file, type, shape=None)</code></p> <p><code>fromstring(datastring, type, shape=None)</code></p> <p>Numeric Types</p> <p><code>A.astype(type)</code> <code>int8, uint8, int16, uint16, int32, uint32,</code> <code>int64, uint64, float32,float64, complex64,</code> <code>complex128</code></p>	<p>are all elements of array nonzero? [also method], identical to <code>alltrue()</code></p> <p>True if all elements within specied amount (between two arrays)</p> <p>Are all elements nonzero along specied axis true.</p> <p>Are any elements of an array nonzero [also method], identical to <code>sometrue()</code></p> <p>Are any elements along specified axis true</p> <p>Find true locations in array <code>a</code></p> <p>Return array with min/max locations for selected axis</p> <p>Returns indices of results of sort on an array</p> <p>Return indices of mapping values of an array <code>a</code> into a monotonic array <code>bin</code></p> <p>Sort array elements along selected axis</p> <p>Fills specied array by selecting corresponding values from a set of arrays using integer selection array (population is a tuple of arrays)</p> <p>Use binary data in file to form new array of specified type.</p> <p>Use binary data in <code>datastring</code> to form new array of specified shape and type</p> <p>Copy of the array, cast to a specified type. Possible numeric types <code>dtype=</code></p>
---	--

PyFits

<pre>import pyfits pyfits.info('pix.fits') img = pyfits.getdata('pix.fits') hdr = pyfits.getheader('pix.fits') hdr['date'] hdr['date'] = '4th of July' hdr.update('flatfile', 'flat17.fits') pyfits.writeto('newfile.fits', data, hdr) pyfits.append('existingfile.fits', data, hdr) pyfits.update('existingfile.fits', data, hdr, ext=3)</pre>	<p>load FITS module</p> <p>show info about file</p> <p>read image data from file</p> <p>read header from file</p> <p>header keyword value</p> <p>modify value</p> <p>add new keyword 'flatfile'</p> <p>create new fits file</p> <p>append to file</p> <p>update file</p>
---	--

Matplotlib/pylab

Plot functions	
<code>acorr(x)</code>	plot autocorrelation function
<code>bar(x,y)</code>	bar charts
<code>barh(x,y)</code>	horizontal bar charts
<code>broken_barh(x,y)</code>	a set of horizontal bars with gaps
<code>boxplot(x)</code>	box and whisker plots
<code>cohere(x,y)</code>	plot of coherence
<code>contour(A)</code>	contour plot
<code>contourf(A)</code>	filled contours
<code>csd(x,y)</code>	plot of cross spectral density
<code>errorbar(x,y,yerr,xerr)</code>	errorbar plot
<code>hist(x)</code>	histogram plot
<code>imshow(A)</code>	display image within axes boundaries (resamples image)
<code>loglog(x,y)</code>	log log plot
<code>matshow(A)</code>	display a matrix in a new figure preserving aspect
<code>pcolor(A)</code>	make a pseudocolor plot
<code>pcolormesh(A)</code>	make a pseudocolor plot using a quadrilateral mesh
<code>pie(x)</code>	pie chart
<code>plot(x,y)</code>	basic x, y plots
<code>plot_date(x,y)</code>	plot using x or y argument as date values and label axis accordingly
<code>polar(x)</code>	polar plot
<code>psd(x)</code>	power spectral density (FFT)
<code>quiver(x,y)</code>	vector field plot
<code>scatter(x,y)</code>	scatter plot
<code>semilogx(x,y)</code>	log x, linear y, x y plot
<code>semilogy(x,y)</code>	linear x, log y, x y plot
<code>specgram</code>	spectrogram plot (FFT)
<code>stem(x,y)</code>	stem plot (similar to histogram)
<code>spy(A)</code>	plot sparsity pattern using markers
<code>xcorr(x,y)</code>	plot the autocorrelation function of x and y

Examples

```
from matplotlib.font_manager import fontManager, FontProperties
font= FontProperties(size='x-small')
pylab.legend(loc='lower left', prop=font)

params = {'backend': 'ps',
          'axes.labelsize': 10,
          'text.fontsize': 10,
          'legend.fontsize': 10,
          'xtick.labelsize': 8,
          'ytick.labelsize': 8,
          'text.usetex': False,
          'figure.figsize': [3.4, 2.1]}
pylab.rcParams.update(params)
```

Based on:

★ *Using Python for Interactive Data Analysis* by Perry Greenfield and Robert Jodrzewski, Space Telescope Science Institute, May 10, 2007

★ *Tentative NumPy Tutorial*, www.scipy.org, March 01, 2009.

Matplotlib/Pylab

Decorations	
annotate	annotate something in figure
arrow	add arrow to plot
axhline	plot horizontal line across axes
axvline	plot vertical line across axes
axhspan	plot horizontal bar across axes
axvspan	plot vertical bar across axes
clabel	label contour lines
clim	adjust color limits of current image
fill	make filled polygons
grid	set whether grids are visible
legend	add legend to current axes
rgrids	customize the radial grids and labels for polar plots
table	add table to axes
text	add text to axes
thetagrids	for polar plots
title	add title to axes
xlabel	add x axes label
ylabel	add y axes label
xlim	set/get x axes limits
ylim	set/get y axes limits
xticks	set/get x ticks
yticks	set/get y ticks
Figure functions	
colorbar	add colorbar to current Figure
figimage	display unresampled image in Figure
figlegend	display legend for Figure
figtext	add text to Figure

Matplotlib/Pylab

Objects	
axes	create axes object on current figure
box	set axis frame state on or off
cla	clear current axes
clf	clear current figure
close	close a figure window
delaxes	delete axes object from the current figure
draw	force a redraw of the current figure
figure	create or change active figure
gca	get the current axes object
gcf	get the current figure
gci	get the current image
getp	get a handle graphics property
hold	set the hold state (overdraw or clear?)
ioff	set interactive mode off
ion	set interactive mode on
isinteractive	test for interactive mode
ishold	test for hold mode
plotting	list plotting commands
rc	control the default parameters
savefig	save the current figure
setp	set a handle graphics property
show	show the current figures (for non-interactive mode)
subplot	create an axes within a grid of axes
subplot_tool	launch the subplot configuration tool
Color Maps	
autumn	set the default colormap to autumn
bone	set the default colormap to bone
cool	set the default colormap to cool
copper	set the default colormap to copper
flag	set the default colormap to flag
gray	set the default colormap to gray
hot	set the default colormap to hot
hsv	set the default colormap to hsv
jet	set the default colormap to jet
pink	set the default colormap to pink
prism	set the default colormap to prism
spring	set the default colormap to spring
summer	set the default colormap to summer
winter	set the default colormap to winter
spectral	set the default colormap to spectral