

Initiation à la programmation en Python

École Normale Supérieure

6 mars 2013

Table des matières

1 scipy

2 pylab

Calcul scientifique: SciPy et Pylab

Syntaxe

```
from scipy import *  
from pylab import *
```

Documentation :

- <http://docs.scipy.org/doc/>
- <http://matplotlib.sourceforge.net/>

Vecteurs et matrices

Type de base : `array`

Vecteur

```
x = array([1,2,3])
```

Matrice

```
M = array([[1,2,3],[4,5,6]])
```

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Construction

Vecteur/Matrice de 1

```
x = ones(5)
M = ones((3,2))
```

$$x = (11111) \quad M = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Matrice identité et matrice diagonale

```
M = eye(3)
N = diag([1,2,3])
```

$$M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad N = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

Construction

Vecteur de valeurs consécutives

```
x = arange(10)
y = linspace(0,1,11)
```

$x = (0123456789)$

$y = (00.10.20.30.40.50.60.70.80.91)$

Opérations

Éléments par éléments

```
x+y  
x-y  
x*y  
x/y  
x**n  
exp(x)  
sqrt(x)
```

Produit scalaire

```
dot(x,y)  
dot(M,x)
```

Opérations

Transposition

`M.T`

Maximum

`M.max()`

Somme

`M.sum()`

Nombre d'éléments

`size(x)`

`M.shape()`

Vecteurs booléens

```
x=array([1,2,3,4])  
x>2.5
```

```
[False, False, True, True]
```

```
A(B)
```

= tous les a_{ij} tels que b_{ij} est True

```
x[x>2.5]+=1
```

ajoute 1 à chaque élément > 2.5

```
M=rand(3,3)  
i,j=where(M>.5)
```

indices (ligne, colonne) des éléments $> .5$

```
i=where(x>2.5)[0]
```

◀ indices des éléments > 2.5 ▶

```
>>> dir(scipy)
```

```
['ALLOW_THREADS', 'BUFSIZE', 'CLIP', 'ERR_CALL', 'ERR_DEFAULT', 'ERR_DEFAULT2', 'ERR_IGNORE', 'ERR_LOG', 'ERR_PRINT', 'ERR_RAISE',  
'ERR_WARN', 'FLOATING_POINT_SUPPORT', 'FPE_DIVIDEBYZERO', 'FPE_INVALID', 'FPE_OVERFLOW', 'FPE_UNDERFLOW', 'False_', 'Inf',  
'Infinity', 'MAXDIMS', 'MachAr', 'NaN', 'NINF', 'NZERO', 'NaN', 'NumpyTest', 'PINF', 'PZERO', 'PackageLoader', 'RAISE', 'RankWarning',  
'True_', 'UFUNC_BUFSIZE_DEFAULT', 'UFUNC_PYVALS_NAME', 'WRAP', '__all__', '__builtins__', '__config__', '__doc__', '__file__',  
'__name__', '__numpy_version__', '__path__', '__version__', 'absolute', 'add', 'add_docstring', 'add_newdoc', 'add_newdocs', 'alen',  
'all', 'allclose', 'alltrue', 'alterdot', 'amax', 'amin', 'angle', 'any', 'append', 'apply_along_axis', 'apply_over_axes', 'arange',  
'arccos', 'arccosh', 'arcsin', 'arcsinh', 'arctan', 'arctan2', 'arctanh', 'argmax', 'argmin', 'argsort', 'argwhere', 'around', 'array',  
'ascontiguousarray', 'asfarray', 'asfortranarray', 'asmatrix', 'asscalar', 'atleast_1d', 'atleast_2d', 'atleast_3d', 'average',  
'bartlett', 'base_repr', 'binary_repr', 'bincount', 'bitwise_and', 'bitwise_not', 'bitwise_or', 'bitwise_xor', 'blackman', 'bmat',  
'bool8', 'bool_', 'broadcast', 'byte', 'c_', 'can_cast', 'cast', 'cdouble', 'ceil', 'central_diff_weights', 'cfloat', 'char', 'character',  
'complex128', 'complex192', 'complex64', 'complex_', 'complexfloating', 'compress', 'concatenate', 'conj', 'conjugate', 'convolve',  
'copy', 'corrcoef', 'correlate', 'cos', 'cosh', 'cov', 'cross', 'csingle', 'ctypeslib', 'cumprod', 'cumproduct', 'cumsum', 'delete',  
'deprecate', 'derivative', 'diag', 'diagflat', 'diagonal', 'diff', 'digitize', 'disp', 'divide', 'dot', 'double', 'dsplit', 'dstack',  
'dtype', 'e', 'ediff1d', 'emath', 'empty', 'empty_like', 'equal', 'errstate', 'exp', 'expand_dims', 'expm1', 'extract', 'eye', 'fabs',  
'factorial', 'factorial2', 'factorialk', 'fastCopyAndTranspose', 'fft', 'fftpack', 'finfo', 'fix', 'flatiter', 'flatnonzero', 'flexible',  
'fromfile', 'fromfunction', 'fromiter', 'frompyfunc', 'fromstring', 'generic', 'get_array_wrap', 'get_include', 'get_numarray_include',  
'get_numpy_include', 'get_printoptions', 'getbuffer', 'getbufsize', 'geterr', 'geterrcall', 'geterrobj', 'gradient', 'greater',  
'greater_equal', 'hamming', 'hanning', 'histogram', 'histogram2d', 'histogramdd', 'hsplit', 'hstack', 'hypot', 'i0', 'identity', 'iff',  
'imag', 'index_exp', 'indices', 'inexact', 'inf', 'info', 'infty', 'inner', 'insert', 'int0', 'int16', 'int32', 'int64', 'int8', 'int_',  
'int_asbuffer', 'intc', 'integer', 'integrate', 'interpolate', 'intersect1d', 'intersect1d_nu', 'intp', 'invert', 'io', 'iscomplex',  
'iscomplexobj', 'isfinite', 'isfortran', 'isinf', 'isnan', 'isneginf', 'isposinf', 'isreal', 'isrealobj', 'isscalar', 'isctype',  
'issubclass', 'issubdtype', 'issubtype', 'iterable', 'ix_', 'kaiser', ' kron', 'ldexp', 'left_shift', 'lena', 'less', 'less_equal',  
'lexsort', 'lib', 'linalg', 'linsolve', 'linspace', 'little_endian', 'load', 'loads', 'log', 'loads', 'log10', 'logip', 'log2', 'logical_and',  
'logical_not', 'logical_or', 'logical_xor', 'logn', 'logspace', 'longdouble', 'longfloat', 'longlong', 'ma', 'mat', 'math', 'matrix',  
'maxentropy', 'maximum', 'maximum_sctype', 'mean', 'median', 'memmap', 'meshgrid', 'mgrid', 'minimum', 'mintypecode', 'misc', 'mod',  
'modf', 'msort', 'multiply', 'nan', 'nan_to_num', 'nanargmax', 'nanargmin', 'nanmax', 'nannin', 'nansum', 'nbytes', 'ndarray',  
'ndenumerate', 'ndim', 'ndimage', 'ndindex', 'negative', 'newaxis', 'newbuffer', 'nonzero', 'not_equal', 'number', 'obj2sctype',  
'object0', 'object_', 'ogrid', 'oldnumeric', 'ones', 'ones_like', 'optimize', 'outer', 'pade', 'pi', 'piecewise', 'pkgload', 'place',  
'poly', 'poly1d', 'polyadd', 'polyder', 'polydiv', 'polyfit', 'polyint', 'polymul', 'polysub', 'polyval', 'power', 'prod', 'product',  
'ptp', 'put', 'putmask', 'r_', 'rand', 'randn', 'random', 'rank', 'ravel', 'real', 'real_if_close', 'rec', 'recarray', 'reciprocal',  
'record', 'remainder', 'repeat', 'require', 'reshape', 'resize', 'restoredot', 'right_shift', 'rint', 'roll', 'rollaxis', 'roots',  
'rot90', 'round', 'row_stack', 's_', 'sctype2char', 'sctypeDict', 'sctypeNA', 'sctypes', 'searchsorted', 'select', 'set_numeric_ops',  
'set_printoptions', 'set_string_function', 'setbufsize', 'setdiffid', 'seterr', 'seterrcall', 'seterrobj', 'setmemberid', 'setxorid',  
'shape', 'short', 'show_config', 'show_numpy_config', 'sign', 'signal', 'signbit', 'signedinteger', 'sin', 'sinc', 'single', 'sinh',  
'size', 'sometrue', 'sort', 'sort_complex', 'source', 'sparse', 'special', 'split', 'sqrt', 'square', 'squeeze', 'stats', 'std', 'str',  
'string0', 'string_', 'subtract', 'sum', 'swapaxes', 'take', 'tan', 'tanh', 'tensordot', 'test', 'tile', 'trace', 'transpose', 'trapz',  
'tri', 'tril', 'trim_zeros', 'triu', 'true_divide', 'typeDict', 'typeNA', 'typecodes', 'typename', 'ubyte', 'ufunc', 'uint', 'uint0',  
'uint16', 'uint32', 'uint64', 'uint8', 'uintc', 'uintp', 'ulonglong', 'unicode0', 'unicode_', 'unionid', 'unique', 'uniqueid',  
'unravel_index', 'unsignedinteger', 'unwrap', 'ushort', 'vander', 'var', 'vdot', 'vectorize', 'version', 'void', 'void0', 'vsplit',  
'vstack', 'where', 'who', 'zeros', 'zeros_like']
```

Tracé de courbes 1D

Le tracé de toutes les courbes "scientifiques" se fait à l'aide de

```
from pylab import *
```

Pour tracer une sinusoïde :

```
x=linspace(-5,5,101) # coordonnées de -5 à 5 avec 101 valeurs  
y=sin(x)  
plot(x,y) # Tracé de la courbe !
```

Pour tracer plusieurs courbes, on peut les mettre les unes à la suite des autres, par exemple :

```
plot(x,y,"r-",x,cos(x),"g.")
```

Tracé de courbes 2D

Pour cela on utilise `imshow(z)` ou `pcolor(x,y,z)`.

```
x=linspace(-5,5,201)
y=linspace(-7,7,201)[ :,newaxis]
    # newaxis indique que ce vecteur est selon la 2ème dimension
z=sin(x**2+y**2)
imshow(z) # Affiche l'image en 2D

imshow(z,extent=(x.min(),x.max(),y.min(),y.max()))
    # On précise les coordonnées des axes

jet() # Pour avoir une plus jolie table de couleurs
```

Diagramme circulaire

Exemple (de Matplotlib)

```
from pylab import *

figure(1, figsize=(6,6))
ax = axes([0.1, 0.1, 0.8, 0.8])

labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
fracs = [15,30,45, 10]
explode=(0, 0.05, 0, 0)
pie(fracs, explode=explode, labels=labels, autopct='%1.1f%%', shadow=True)
title('Raining Hogs and Dogs', bbox={'facecolor':'0.8', 'pad':5})
show()
```

Diagramme circulaire

