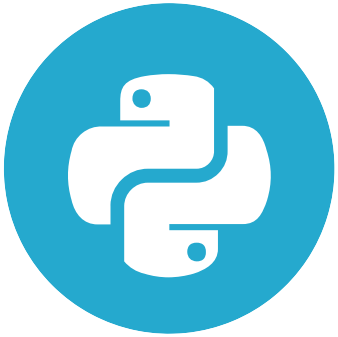


SciPy Cheat Sheet: Linear Algebra in Python

February 7th, 2017 in Python

Karlijn Willems

By now, you will have already learned that NumPy, one of the fundamental packages for scientific computing, forms at least for a part the fundament of other important packages that you might use used for data manipulation and machine learning with Python. One of those packages is SciPy, another one of the core packages for scientific computing in Python that provides mathematical algorithms and convenience functions built on the NumPy extension of Python.

You might now wonder why this library might come in handy for data science.

Well, SciPy has many modules that will help you to understand some of the basic components that you need to master when you're learning data science, namely, math, stats and machine learning. You can find out what other things you need to tackle to learn data

you're looking for to understand machine learning concepts such as eigenvalues, regression, and r

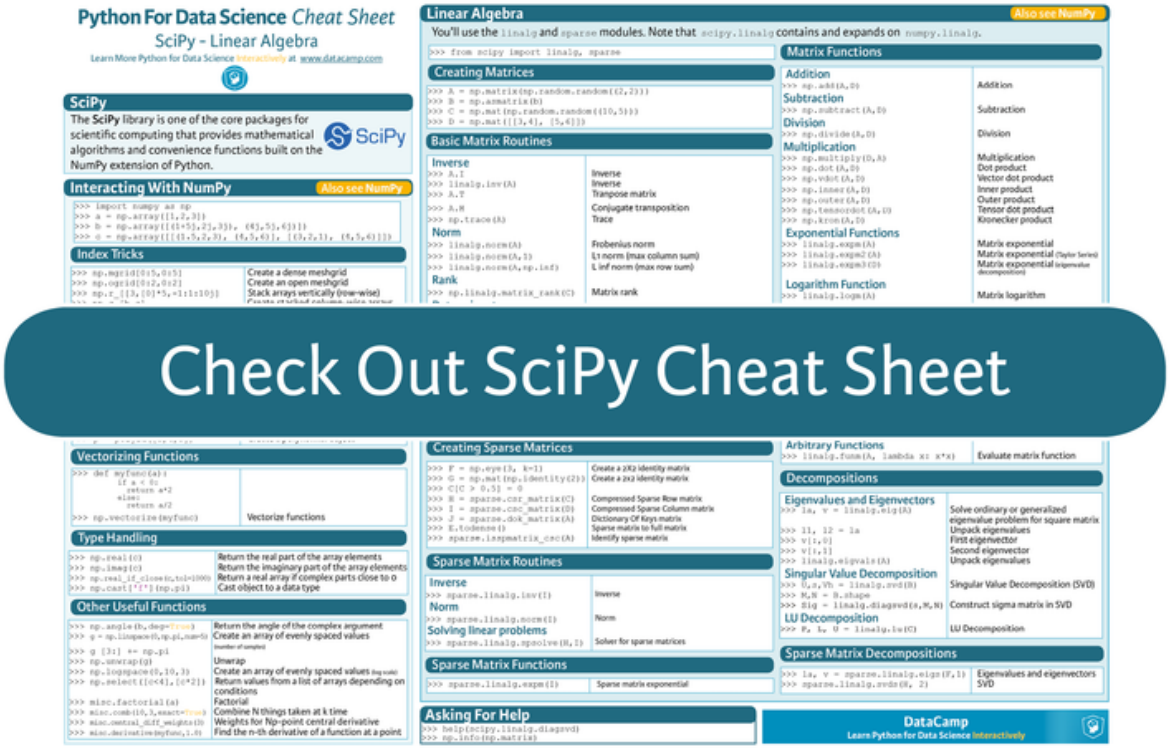
But, what is maybe the most obvious is that most machine learning techniques deal with high-dime
represented as matrices. What's more, you'll need to understand how to manipulate these matrices

learners and start one of our interactive tutorials today!

Learn R Learn Python

That is why DataCamp has made a SciPy cheat sheet that will help you to master linear algebra with Python.

Take a look by clicking on the button below:



such as matrix creation, matrix functions, basic routines that you can perform with matrices, and m
`scipy.linalg` . Sparse matrices are also included, with their own routines, functions, and decomposit
module.

learners and start one of our interactive tutorials today!

Learn R Learn Python

(Above is the printable version of this cheat sheet)

Python for Data-Science *Cheat Sheet*: SciPy - Linear Algebra

SciPy

The SciPy library is one of the core packages for scientific computing that provides mathematical algorithms and convenience functions built on the NumPy extension of Python.

Asking For Help

```
>>> help(scipy.linalg.diagsvd)
```

Interacting With NumPy

```
>>> import numpy as np
>>> a : np.array([1,2,3])
>>> b : np.array([(1+5j,2j,3j), (4j,5j,6j)])
>>> c : np.array([[(1.5,2,3), (4,5,6)], [(3,2,1), (4,5,6)]])
```

Index Tricks

```
>>> np.mgrid[0:5,0:5]
```

Create a dense meshgrid

```
>>> np.c_[b,c]
```

learners and start one of our interactive tutorials today!

Learn R Learn Python

Shape Manipulation

```
>>> np.transpose(b)
```
Permute array dimensions

```
>>> b.flatten()
```
Flatten the array

```
>>> np.hstack((b,c))
```
Stack arrays horizontally (column-wise)

```
>>> np.vstack((a,b))
```
Stack arrays vertically (row-wise)

```
>>> np.hsplit(c,2)
```
Split the array horizontally at the 2nd index

```
>>> np.vpslit(d,2)
```

Polynomials

```
>>> from numpy import poly1d
```

```
>>> p : poly1d([3,4,5])
```
Create a polynomial object

Vectorizing Functions

```
>>> def myfunc(a):
      if a < 0:
      return a*2
      else:
      return a/2
```

Type Handling

`>>> np.real(b)`	Return the real part of the array
`>>> np.imag(b)`	Return the imaginary part of the
`>>> np.real_if_close(c,tol:1000)`	Return a real array if complex parts close to 0
`>>> np.cast['f'](np.pi)`	Cast object to a data type

learners and start one of our interactive tutorials today!

Learn R Learn Python

Other Useful Functions

`>>> np.angle(b,deg:True)`	Return the angle of the complex argument
`>>> g : np.linspace(0,np.pi,num:5)`	Create an array of evenly spaced values (number of samples)
`>>> g [3:] +: np.pi`	
`>>> np.unwrap(g)`	
`>>> np.logspace(0,10,3)`	Create an array of evenly spaced values (log scale)
`>>> np.select([c<4],[c*2])`	Return values from a list of arrays depending on conditions
`>>> misc.factorial(a)`	Factorial
`>>> misc.comb(10,3,exact:True)`	Combine N things taken at k time
`>>> misc.central_diff_weights(3)`	Weights for Np-point central derivative
`>>> misc.derivative(myfunc,1.0)`	Find the n-th derivative of a function at a point

Linear Algebra

You'll use the linalg and sparse modules. Note thatscipy.linalg contains and expands onnumpy.linalg.

Creating Matrices

```
>>> A : np.matrix(np.random.random((2,2)))
```

```
>>> B : np.asmatrix(b)
```

```
>>> C : np.mat(np.random.random((10,5)))
```

```
>>> D : np.mat([[3,4], [5,6]])
```

learners and start one of our interactive tutorials today!

Learn R Learn Python

Basic Matrix Routines

Inverse

```
>>> A.I
```
Inverse

```
>>> linalg.inv(A)
```
Inverse

Transposition

```
>>> A.T
```
Tranpose matrix

```
>>> A.H
```
Conjugate transposition

Trace

```
>>> np.trace(A)
```
Trace

Norm

```
>>> linalg.norm(A)
```
Frobenius norm

```
>>> linalg.norm(A,1)
```
L1 norm (max column sum)

Rank

`>>> np.linalg.matrix_rank(C)`	Matrix rank

learners and start one of our interactive tutorials today!

Learn R Learn Python

Determinant

`>>> linalg.det(A)`	Determinant

Solving linear problems

`>>> linalg.solve(A,b)`	Solver for dense matrices
`>>> E : np.mat(a).T`	Solver for dense matrices
`>>> linalg.lstsq(F,E)`	Least-squares solution to linear matrix equation

Generalized inverse

`>>> linalg.pinv(C)`	Compute the pseudo-inverse of a matrix (least-squares solver
`>>> linalg.pinv2(C)`	Compute the pseudo-inverse of a matrix (SVD)

Creating Sparse Matrices

`>>> F : np.eye(3, k:1)`	Create a 2X2 identity matrix
`>>> G : np.mat(np.identity(2))`	Create a 2x2 identity matrix
`>>> C[C > 0.5] : 0`	
`>>> H : sparse.csr_matrix(C)`	Compressed Sparse Row matrix
`>>> I : sparse.csc_matrix(D)`	Compressed Sparse Column matrix
`>>> J : sparse.dok_matrix(A)`	Dictionary Of Keys matrix

learners and start one of our interactive tutorials today!

Learn R Learn Python

Sparse Matrix Routines

Inverse

```
>>> sparse.linalg.inv(I)
```

Inverse

Norm

```
>>> sparse.linalg.norm(I)
```

Norm

Solving linear problems

```
>>> sparse.linalg.spsolve(H,I)
```

Solver for sparse matrices

Sparse Matrix Functions

```
>>> la, v : sparse.linalg.eigs(F,1)
```

Eigenvalues and eigenvectors

```
>>> sparse.linalg.svds(H, 2)
```

SVD

```
>>> sparse.linalg.expm(I)
```

Sparse matrix exponential

Matrix Functions

Addition

```
>>> np.add(A,D)
```

Addition

Subtraction

Division

`>>> np.divide(A,D)`	Division

learners and start one of our interactive tutorials today!

Learn R Learn Python

Multiplication

`>>> A @ D`	Multiplication operator (Python 3)
`>>> np.multiply(D,A)`	Multiplication
`>>> np.dot(A,D)`	Dot product
`>>> np.vdot(A,D)`	Vector dot product
`>>> np.inner(A,D)`	Inner product
`>>> np.outer(A,D)`	Outer product
`>>> np.tensordot(A,D)`	Tensor dot product
`>>> np.kron(A,D)`	Kronecker product

Exponential Functions

`>>> linalg.expm(A)`	Matrix exponential
`>>> linalg.expm2(A)`	Matrix exponential (Taylor Series)
`>>> linalg.expm3(D)`	Matrix exponential (eigenvalue decomposition)

Logarithm Function

`>>> linalg.logm(A)`	Matrix logarithm

Trigonometric Functions

`>>> linalg.tanm(A)`	Matrix tangent

learners and start one of our interactive tutorials today!

Learn R Learn Python

Hyperbolic Trigonometric Functions

`>>> linalg.sinhm(D)`	Hypberbolic matrix sine
`>>> linalg.coshm(D)`	Hyperbolic matrix cosine
`>>> linalg.tanhm(A)`	Hyperbolic matrix tangent

Matrix Sign Function

`>>> np.signm(A)`	Matrix sign function

Matrix Square Root

`>>> linalg.sqrtm(A)`	Matrix square root

Arbitrary Functions

`>>> linalg.funm(A, lambda x: x*x)`	Evaluate matrix function

Decompositions

Eigenvalues and Eigenvectors

`>>> la, v : linalg.eig(A)`	Solve ordinary or generalized eigenvalue problem for square matrix
`>>> l1, l2 : la`	Unpack eigenvalues
`>>> v[:,0]`	First eigenvector

learners and start one of our interactive tutorials today!

Learn R Learn Python

Singular Value Decomposition

LU Decomposition

`>>> U,s,Vh : linalg.svd(B)`	Singular Value Decomposition (SVD)
`>>> M,N : B.shape`	
`>>> Sig : linalg.diagsvd(s,M,N)`	Construct sigma matrix in SVD
`>>> P,L,U : linalg.lu(C)`	LU Decomposition

Sparse Matrix Decompositions

```
>>> np.info(np.matrix)
```

PS. Don't miss our other Python cheat sheets for data science that cover Numpy, Scikit-Learn, Bokeh, Pandas and the Python basics.

What do you think?

Python

Up Next

learners and start one of our interactive tutorials today!

New Course! Supervised Learning in R: Classification

Learn R **Learn Python**

R Programming

79 views

This beginner-level introduction to machine learning covers four of the most common classification algorithms. You will come away with a b…

September 27th, 2017 in Blog

New Course: Case Studies in Statistical Thinking!

Python

1,139 views

Hone your applied data science skills in Python by doing a variety of case studies across multiple disciplines. Use data science to solve …

September 20th, 2017 in Blog

learners and start one of our interactive tutorials today!

Jupyter Notebook Cheat Sheet

Learn R **Learn Python**

Python 19,148 views

This Jupyter Notebook cheat sheet will help you to find your way around the well-known Jupyter Notebook App, a subproject of Project Jupyt…

September 19th, 2017 in Blog

View All

Comments

danieltemkin

Thanks for the cheat sheet. It will hopefully prevent me from needing to keep a scipy browser tab open. Just wondering if you have checked out Sympy? I rarely see it mentoned but in my opinion it is one of the best packages out there in terms of building anonymous math functions and it handles complex math concepts like fourier transforms, taylor series, integrals, differentiable equations, and many many more. Check it out when you get a chance.

02/15/17 3:16 AM |

karlijn

Hi Daniel, thank you very much for your interest and feedback! I have already heard of Sympy, but have never used it, so I'll make sure to check this out! Thanks!

02/21/17 1:52 PM |

Sign In to Comment