
PYTHONによるデータサイエンス

Pythonの入門と橋渡し

用いるリンク先

- <https://www-bcf.usc.edu/~gareth/ISL/>. (<http://bit.ly/ISLRusc>)
- <https://www.r-bloggers.com/in-depth-introduction-to-machine-learning-in-15-hours-of-expert-videos/> (<http://bit.ly/ISLRrbloggers>)
- <https://github.com/JWarmenhoven/ISLR-python> (<http://bit.ly/ISLRpython>)

今回は

- Introduction to Statistical Learning with Applications in Rは良い本です
- 最近、Rによる統計的学習入門と題した訳書が朝倉書店から出版されました
- この本はすべてがRのコードで用意されていますが、好事家が大半をPythonで実行できるようにコードを用意してくれています
(JWarmenhoven,jcrouser,mscaudill,sujitpal)
- 前回は、柔軟な学習機械ほど、複雑な問題ほど、大きなデータを必要とすること、当てはまりの良さよりは汎化能力が大事なこと、教師あり学習と教師なし学習について述べました。今回はClassificationとResampling Methodsを見ていこうと思います。

今回は

- Pythonのおさらいを少ししてから、またPythonのコードを眺めていこうと思います

PYTHONの復習

- Pythonはプログラミング言語と処理系の双方の名称
- Pythonのプログラムコードを実行するにはvi(m)、emacs、atomなど、自分の好みのテキストエディタで次の一行を入力し、hello.pyというテキストファイルに保存する

```
print "Hello, World!"
```

- そしてpythonというコマンドに、そのファイルを引数で与えてあげる

```
> python hello.py
```

(“>”はコマンドプロンプト)

- 拡張子.pyを持つファイルとpythonという名前のアプリケーションを連携させる機能をもつOSなら、ダブルクリックするだけでいい
- ループや関数の中身など、階層構造を字下げ（タブ）で表す

PYTHONの使い方

- 準備

- Anaconda、本家のPythonインストーラ、ActiveStateのPythonインストーラ、Linuxディストリビューションのパッケージ、HomeBrew、MacPortsなどから、どれか一つをインストールします
- pip、pyenv、virtualenvが便利と聞いた人は、それらの指示に従ってください

- 実行

- pythonを実行して、コードをコピペする（最初だけ）
- プログラムコードを記したソースファイルを作成し、pythonの実行時に読み込ませる（本番はこれが便利）
- Jupyter[Hub, Notebook]を通してipythonを使う（学習時はこれが便利）

PYTHONはデータ型が少ない

- Pythonは圧倒的に標準のデータ型も少ない
- 論理型はbool
- 数値型はint(整数), float(浮動小数点), complex(複素数)の3種類
- 文字列型はstr (文字型はない)
- 配列はlist(リスト、[1,2]), tuple((1,2)), range、set(集合)、dict(辞書)
- 他にb bytearray、bitsなどもある
- たとえば行列が標準ではないなど、ミニマルな設計方針のよう

PYTHONは予約語も少ない

- Pythonは圧倒的に予約語が少ないです

Pythonの予約語一覧

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

- これらは自分の変数や関数に使えません
- 言語自身について覚えることが少ないので、使用目的に早めに専念できます

何かにつけてIMPORT

- Python本体は極めてミニマル
- モジュールは拡張子.pyを持つファイルで、Python言語で記述されたオブジェクトやメソッドの定義が入っています
- モジュールは、importすると利用可能になります

```
import numpy as np;
```

- これでNumPyのモジュールnumpyを、npという名前でimportします
- パッケージは、__init__.pyというファイルを持つ複数のモジュールを収めたフォルダです

```
import matplotlib.pyplot as plt;
```

- これで、Matplotlibというパッケージの中のPyPlotというモジュールをimportします

FROMも使う

- Matplotlibというパッケージの中のPyPlotというモジュールをimportするには、次の2つのやり方があります

```
import matplotlib.pyplot as plt;
```

```
from matplotlib import pyplot as plt;
```

- どちらでもいいです

IMPORTの仕方

- モジュールのimportの仕方は少なくとも次の4種類がある

```
import numpy;  
numpy.array([1,2,3,4]);
```

```
from numpy import *;  
array([1,2,3,4]);
```

```
import numpy as np;  
np.array([1,2,3,4]);
```

```
from numpy import array;  
array([1,2,3,4]);
```

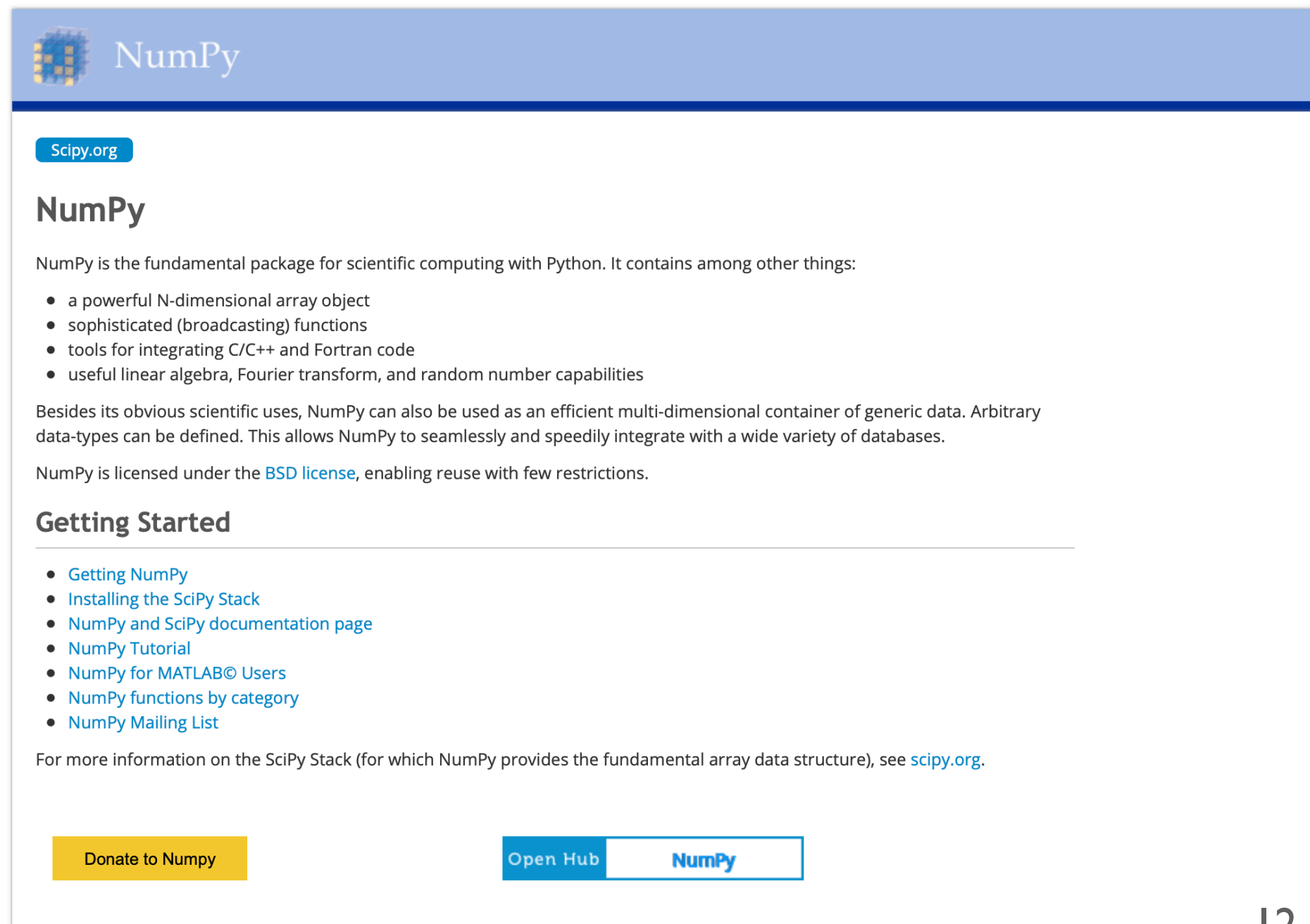
- どれを用いても良い
- importした関数や変数などの名前が「衝突」することがある
- 衝突を回避するには、importした関数や変数にモジュール名かモジュール名のイニシャルを付けて使う方法が望ましい
- 小規模のコードであれば、気にしなくていい

NUMPY

- NumPyは数値計算のためのモジュール群
- 多次元配列ndarrayとその操作、また配列同士や配列を用いる数学関数を提供する
- NumPyが提供する配列arrayが重要

```
import numpy as np;
A = np.array([[1,2,3],[4,5,6]]);
B = np.array([[1,2],[3,4],[5,6]]);
print A;
print A.dot(B);
```

- 行列matrixもあり、行列の和・差・積も可能
- 円周率pi、Napier数e、Eulerの γ euler_gammaもここで定義される



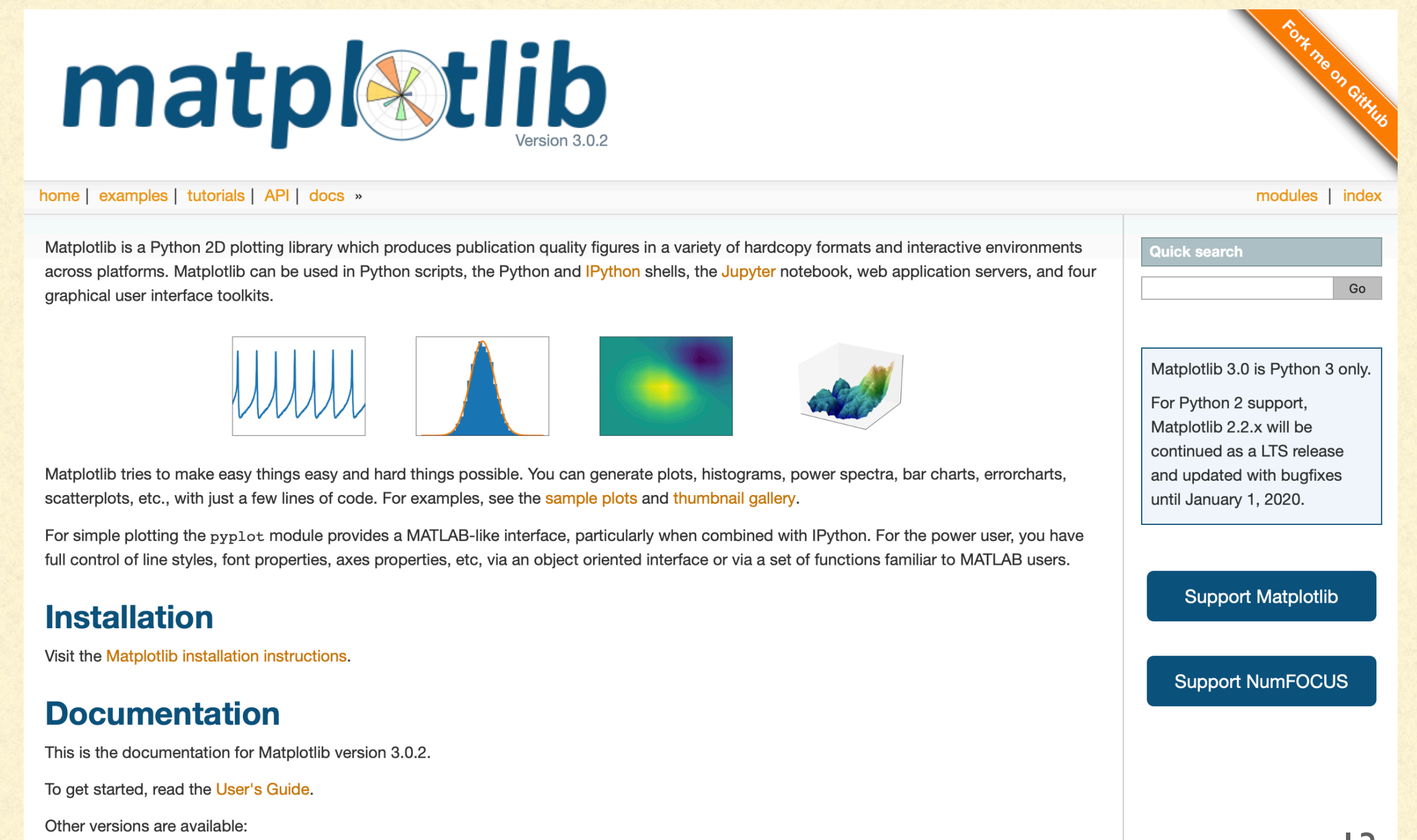
The screenshot shows the NumPy page on Scipy.org. At the top, there is a blue header with the NumPy logo and the text "NumPy". Below the header, there is a blue button that says "Scipy.org". The main heading is "NumPy". The text below the heading states: "NumPy is the fundamental package for scientific computing with Python. It contains among other things:" followed by a bulleted list: "a powerful N-dimensional array object", "sophisticated (broadcasting) functions", "tools for integrating C/C++ and Fortran code", and "useful linear algebra, Fourier transform, and random number capabilities". Below this, it says: "Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases." and "NumPy is licensed under the [BSD license](#), enabling reuse with few restrictions." There is a section titled "Getting Started" with a list of links: "Getting NumPy", "Installing the SciPy Stack", "NumPy and SciPy documentation page", "NumPy Tutorial", "NumPy for MATLAB Users", "NumPy functions by category", and "NumPy Mailing List". At the bottom, there is a yellow button "Donate to Numpy" and a blue button "Open Hub" with the NumPy logo next to it.

MATPLOTLIB

- Matplotlibはデータの可視化モジュール
- 多くの2次元グラフを提供する (ギャラリーを各自で見てくださいと良い)

```
import matplotlib.pyplot as plt;
```

```
from matplotlib import pyplot as plt;
```



matplotlib Version 3.0.2

home | examples | tutorials | API | docs » modules | index

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For examples, see the [sample plots](#) and [thumbnail gallery](#).

For simple plotting the `pyplot` module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Installation

Visit the [Matplotlib installation instructions](#).

Documentation

This is the documentation for Matplotlib version 3.0.2.

To get started, read the [User's Guide](#).

Other versions are available:

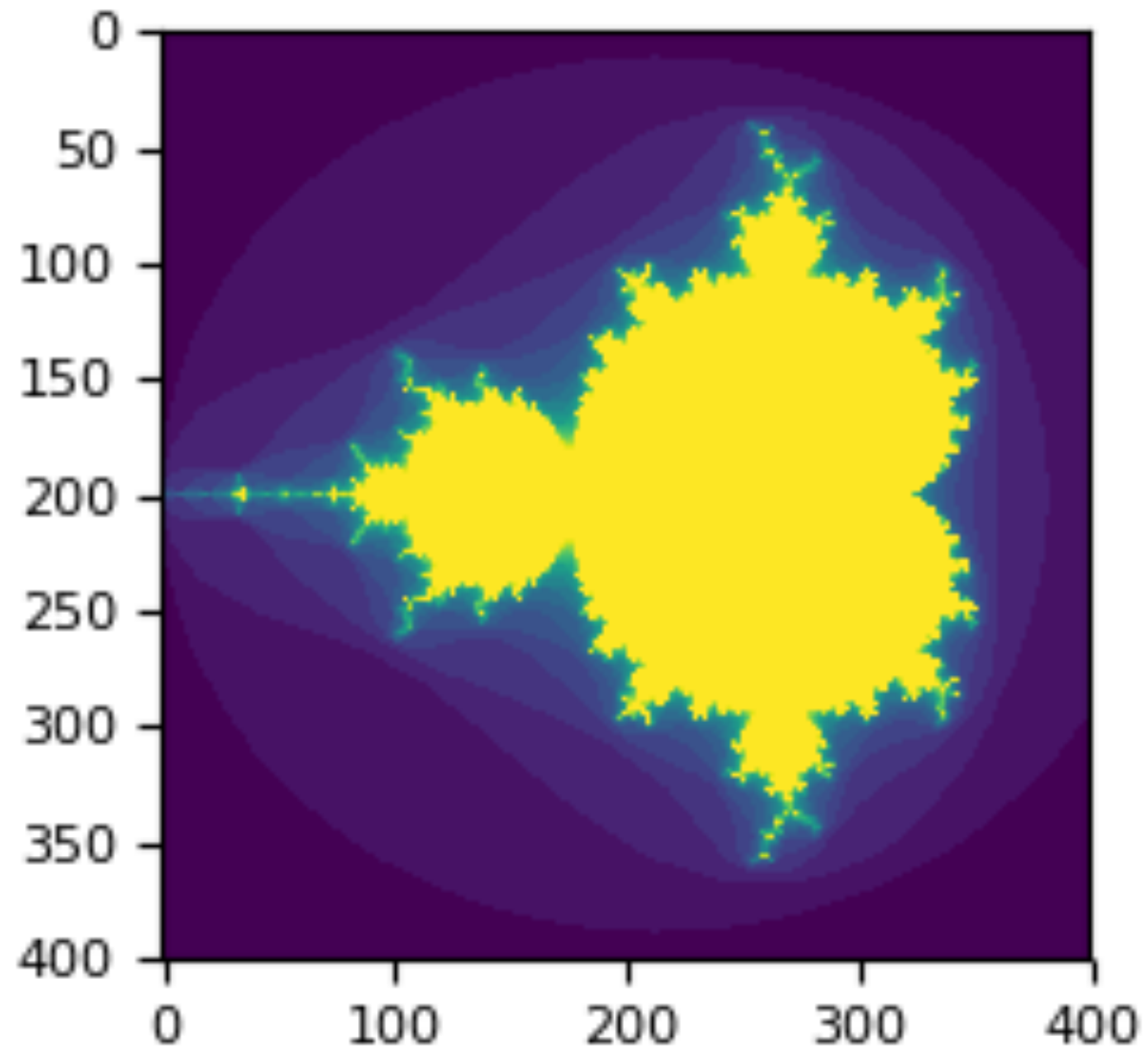
Quick search: Go

Matplotlib 3.0 is Python 3 only. For Python 2 support, Matplotlib 2.2.x will be continued as a LTS release and updated with bugfixes until January 1, 2020.

Support Matplotlib

Support NumFOCUS

NUMPYとMATPLOTLIBの例



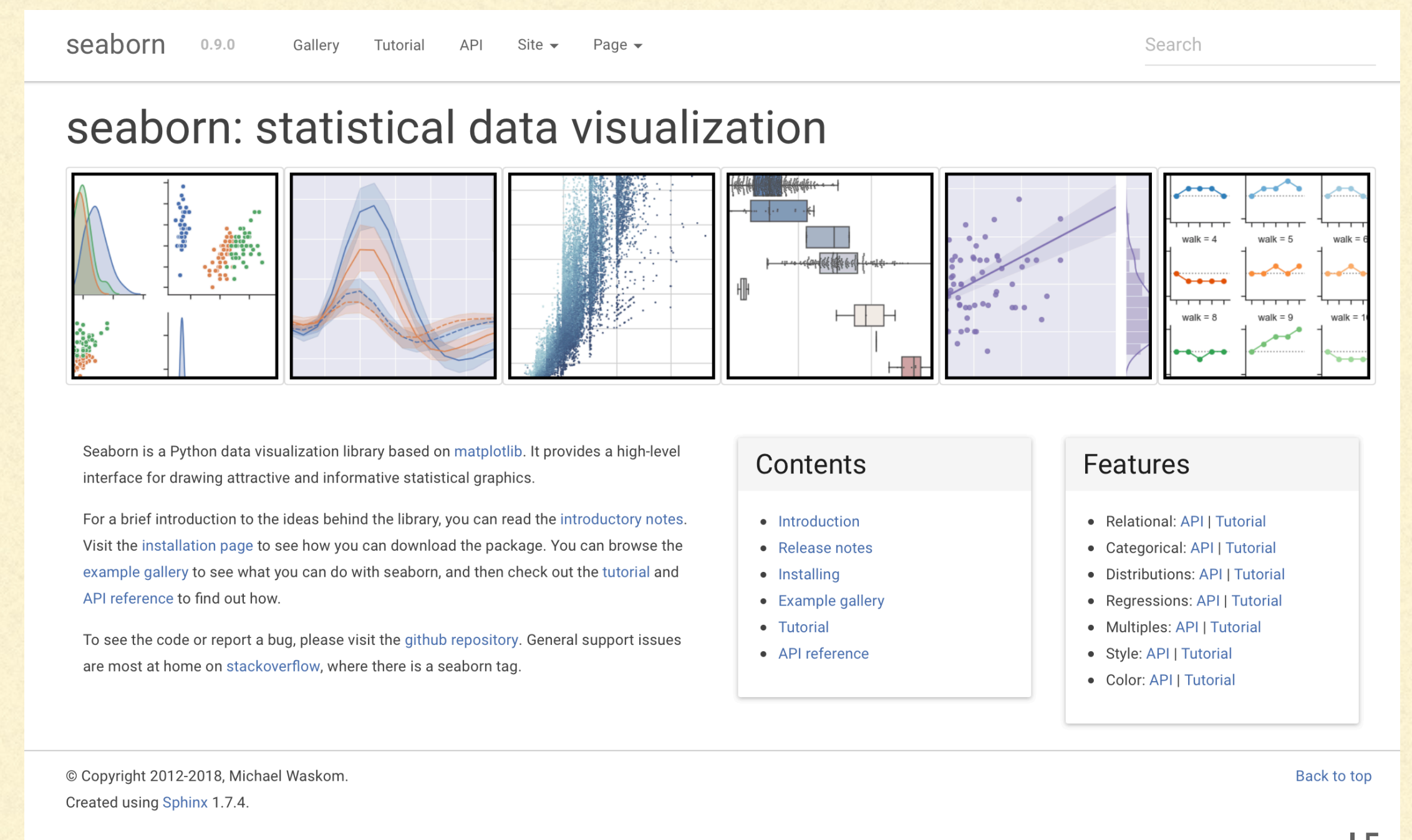
```
import numpy as np;
import matplotlib.pyplot as plt;
def mandelbrot( h,w, maxit=20 ):
    """Returns an image of the Mandelbrot fractal of size (h,w)."""
    y,x = np.ogrid[ -1.4:1.4:h*1j, -2:0.8:w*1j ]
    c = x+y*1j
    z = c
    divtime = maxit + np.zeros(z.shape, dtype=int)

    for i in range(maxit):
        z = z**2 + c
        diverge = z*np.conj(z) > 2**2 # who is diverging
        div_now = diverge & (divtime==maxit) # who is diverging now
        divtime[div_now] = i # note when
        z[diverge] = 2 # avoid diverging too much

    return divtime
plt.imshow(mandelbrot(400,400));
plt.show();
```

SEABORN

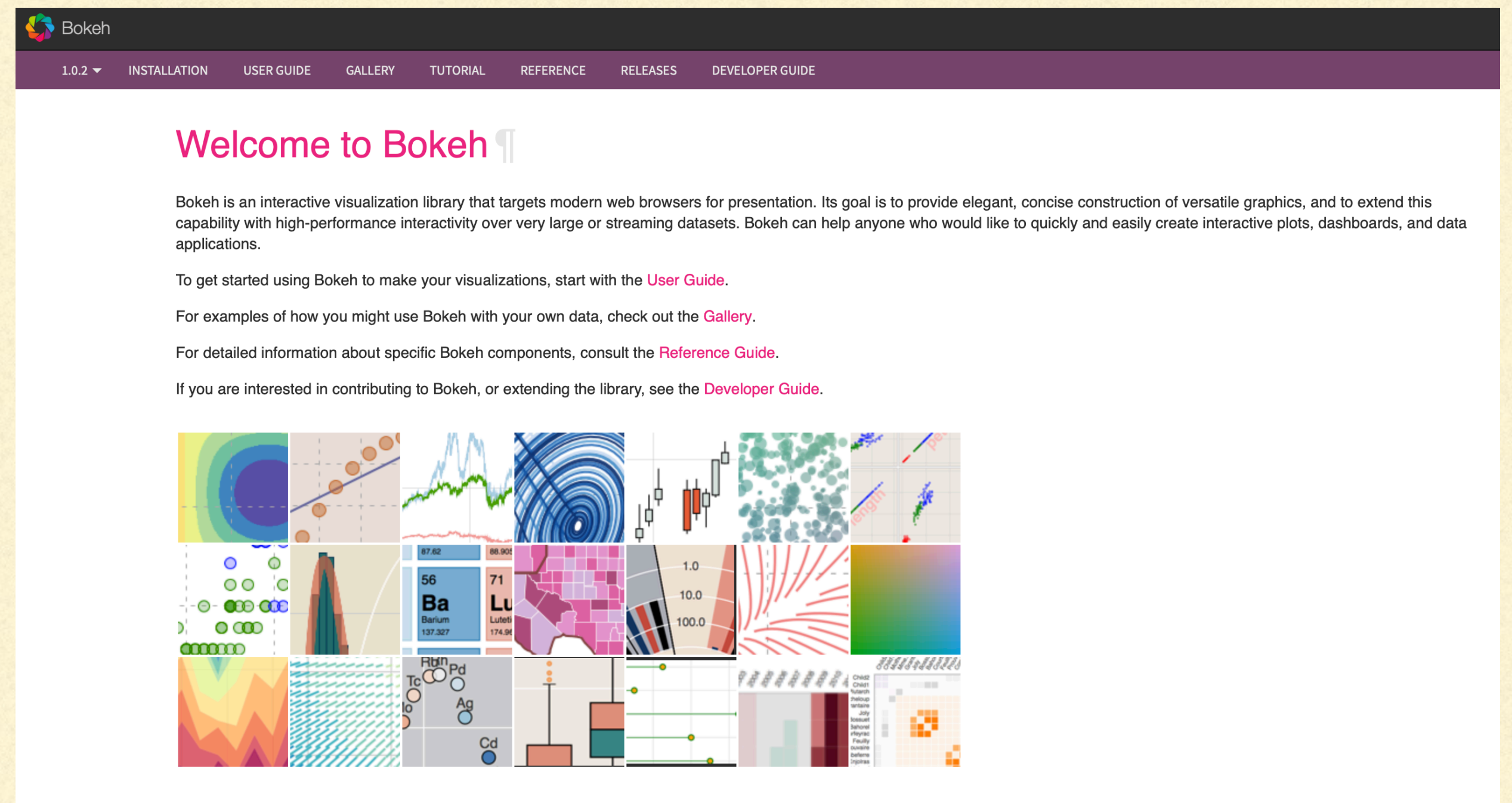
- Seabornも統計的なデータの可視化モジュールで、Matplotlibをベースに開発されています
- これもギャラリーを見ていただくと良いです



The screenshot shows the Seaborn website interface. At the top, there is a navigation bar with links for "seaborn", "0.9.0", "Gallery", "Tutorial", "API", "Site", and "Page". A search bar is located on the right. The main heading is "seaborn: statistical data visualization". Below this, there is a grid of six thumbnail images showing different types of plots: a density plot, a scatter plot with regression lines, a box plot, a scatter plot with a regression line, and a grid of line plots. Below the grid, there is a "Contents" section with links to "Introduction", "Release notes", "Installing", "Example gallery", "Tutorial", and "API reference". To the right of the "Contents" section is a "Features" section with links to "Relational: API | Tutorial", "Categorical: API | Tutorial", "Distributions: API | Tutorial", "Regressions: API | Tutorial", "Multiples: API | Tutorial", "Style: API | Tutorial", and "Color: API | Tutorial". At the bottom left, there is a copyright notice: "© Copyright 2012-2018, Michael Waskom. Created using Sphinx 1.7.4." At the bottom right, there is a "Back to top" link.

BOKEH

- Bokehは、ブラウザを用いたインタラクティブな可視化を実現するモジュールです
- Bokehサーバを経由します
- こちらのギャラリーは素晴らしいの一言です



The screenshot shows the Bokeh website homepage. At the top, there is a navigation bar with the Bokeh logo and links for 1.0.2, INSTALLATION, USER GUIDE, GALLERY, TUTORIAL, REFERENCE, RELEASES, and DEVELOPER GUIDE. Below the navigation bar, the main content area features a heading "Welcome to Bokeh" followed by a paragraph describing Bokeh as an interactive visualization library. Below this, there are three lines of text with links to the User Guide, Gallery, and Reference Guide. At the bottom, there is a large grid of various data visualizations, including scatter plots, line graphs, heatmaps, and histograms.

SCIPLY

- SciPyはPythonによる数学、科学、工学のためのオープンソースソフトウェアのエコシステム（要するに科学技術計算のためのパッケージ群）でNumPyも含まれる
- Python, NumPy, SciPyライブラリ, Matplotlibの上に構築されている

SciPy.org

Install Getting Started Documentation Report Bugs Blogs

SciPy (pronounced "Sigh Pie") is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

- NumPy: Base N-dimensional array package
- SciPy library: Fundamental library for scientific computing
- Matplotlib: Comprehensive 2D Plotting
- IPython: Enhanced Interactive Console
- Sympy: Symbolic mathematics
- pandas: Data structures & analysis

More information...

News

- NumPy 1.15.4 released (2018-11-04) See [Obtaining NumPy & SciPy libraries](#).
- NumPy 1.15.3 released (2018-10-22) See [Obtaining NumPy & SciPy libraries](#).

Scientific Computing Tools for Python

SciPy refers to several related but distinct entities:

- The *SciPy ecosystem*, a collection of open source software for scientific computing in Python.
- The *community* of people who use and develop this stack.
- Several *conferences* dedicated to scientific computing in Python - SciPy, EuroSciPy and SciPy.in.
- The *SciPy library*, one component of the SciPy stack, providing many numerical routines.

The SciPy ecosystem

Scientific computing in Python builds upon a small core of packages:

- **Python**, a general purpose programming language. It is interpreted and dynamically typed and is very suited for interactive work and quick prototyping, while being powerful enough to write large applications in.
- **NumPy**, the fundamental package for numerical computation. It defines the numerical array and matrix types and basic operations on them.
- The **SciPy library**, a collection of numerical algorithms and domain-specific toolboxes, including signal processing, optimization, statistics and much more.
- **Matplotlib**, a mature and popular plotting package, that provides publication-quality 2D plotting as well as rudimentary 3D plotting

On this base, the SciPy ecosystem includes general and specialised tools for data management and computation, productive experimentation and high-performance computing. Below we overview some key packages, though there are [many more relevant packages](#).

Data and computation:

- **pandas**, providing high-performance, easy to use data structures.
- **SymPy**, for symbolic mathematics and computer algebra.
- **scikit-image** is a collection of algorithms for image processing.
- **scikit-learn** is a collection of algorithms and tools for machine learning.
- **h5py** and **PyTables** can both access data stored in the HDF5 format.

Productivity and high-performance computing:

- **IPython**, a rich interactive interface, letting you quickly process data and test ideas.
- The **Jupyter** notebook provides IPython functionality and more in your web browser, allowing you to document your computation in an easily reproducible form.
- **Cython** extends Python syntax so that you can conveniently build C extensions, either to speed up critical code, or to integrate with C/C++ libraries.
- **Dask**, **Joblib** or **IPyParallel** for distributed processing with a focus on numeric data.

Quality assurance:

- **nose**, a framework for testing Python code, being phased out in preference for **pytest**.
- **numpydoc**, a standard and library for documenting Scientific Python libraries.

About SciPy

- SciPy Stack
- Getting Started
- Documentation
- Install
- Bug Reports
- Codes of Conduct
- SciPy Conferences
- Topical Software
- Citing
- Cookbook
- Blogs
- NumFOCUS

CORE PACKAGES:

- Numpy
- SciPy library
- Matplotlib
- IPython
- Sympy
- Pandas

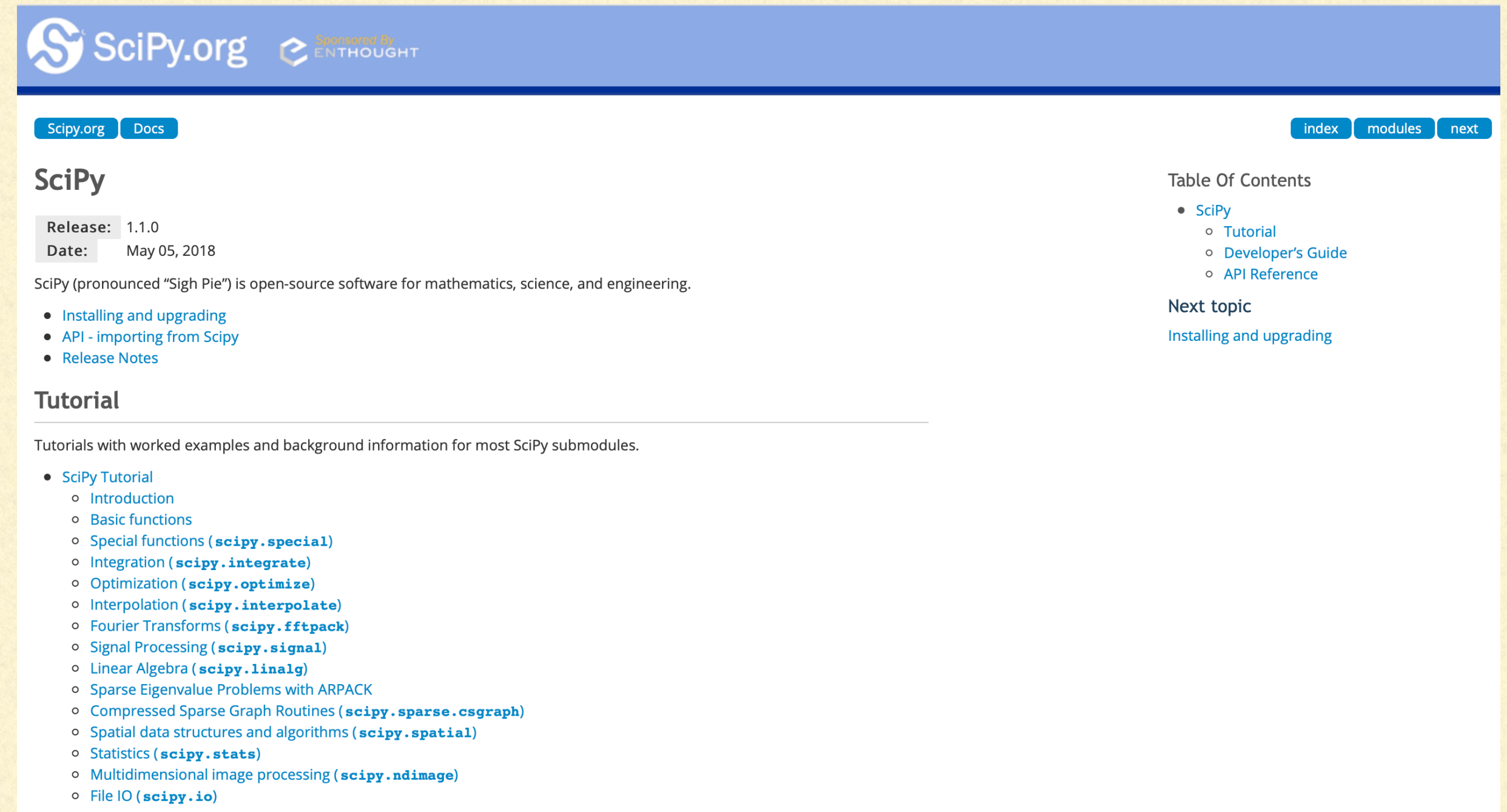
Table Of Contents

- Scientific Computing Tools for Python
 - The SciPy ecosystem

Search

SCIPY

- ライブラリもSciPy (SciPy libraryと呼び分けている)
- クラスタリングcluster、定数constants、離散フーリエ変換fftpack、積分と常微分方程式integrate、補間interpolate、入出力io、線形代数linalg、その他misc、多次元画像処理ndimage、直交距離回帰odr、最適化optimize、信号処理signal、疎行列sparse、疎行列の固有値問題、疎グラフsparse.csgraph、計算幾何spatial、特殊関数special、確率・統計stats、C言語とC++言語のインタフェースweave

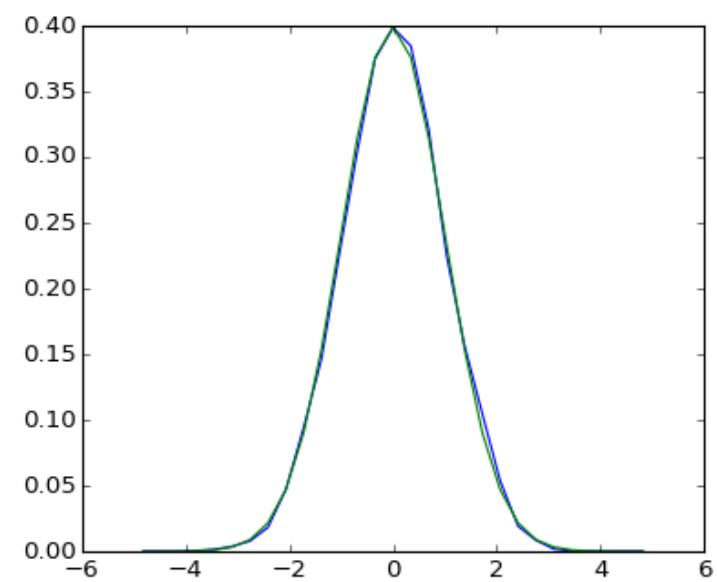
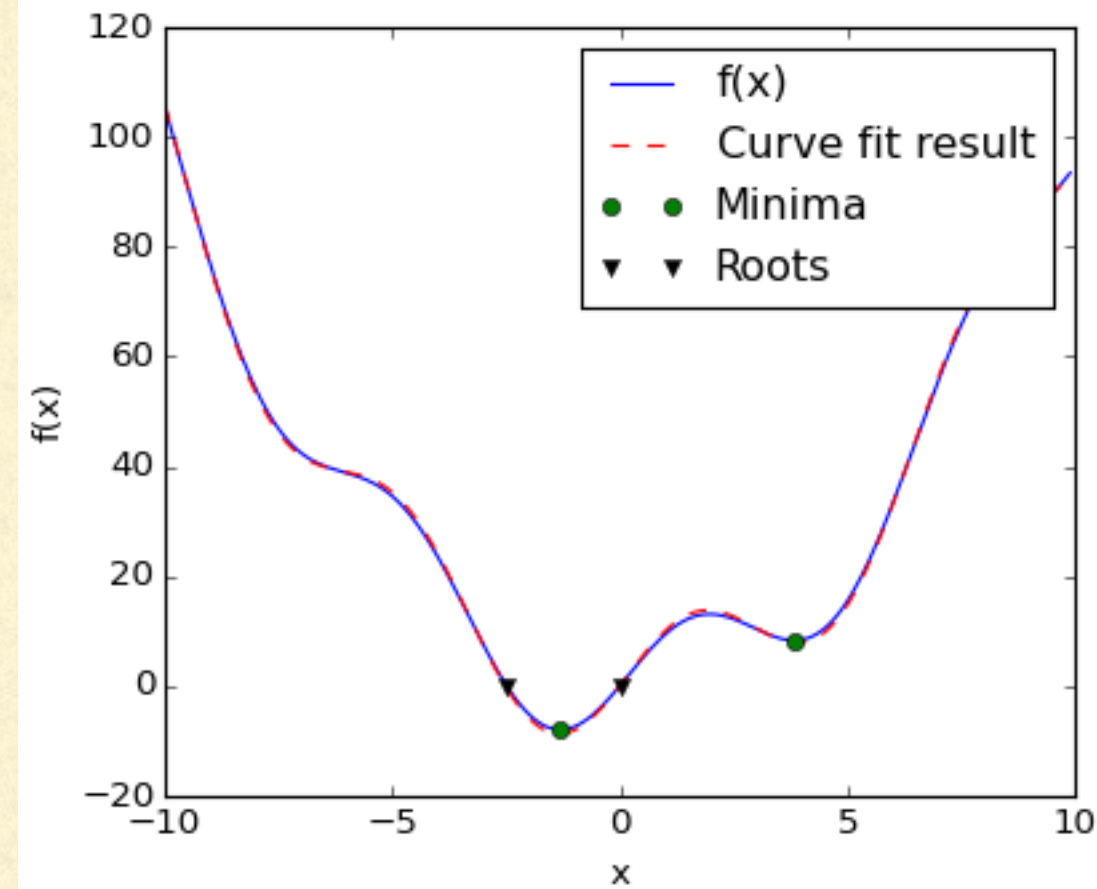
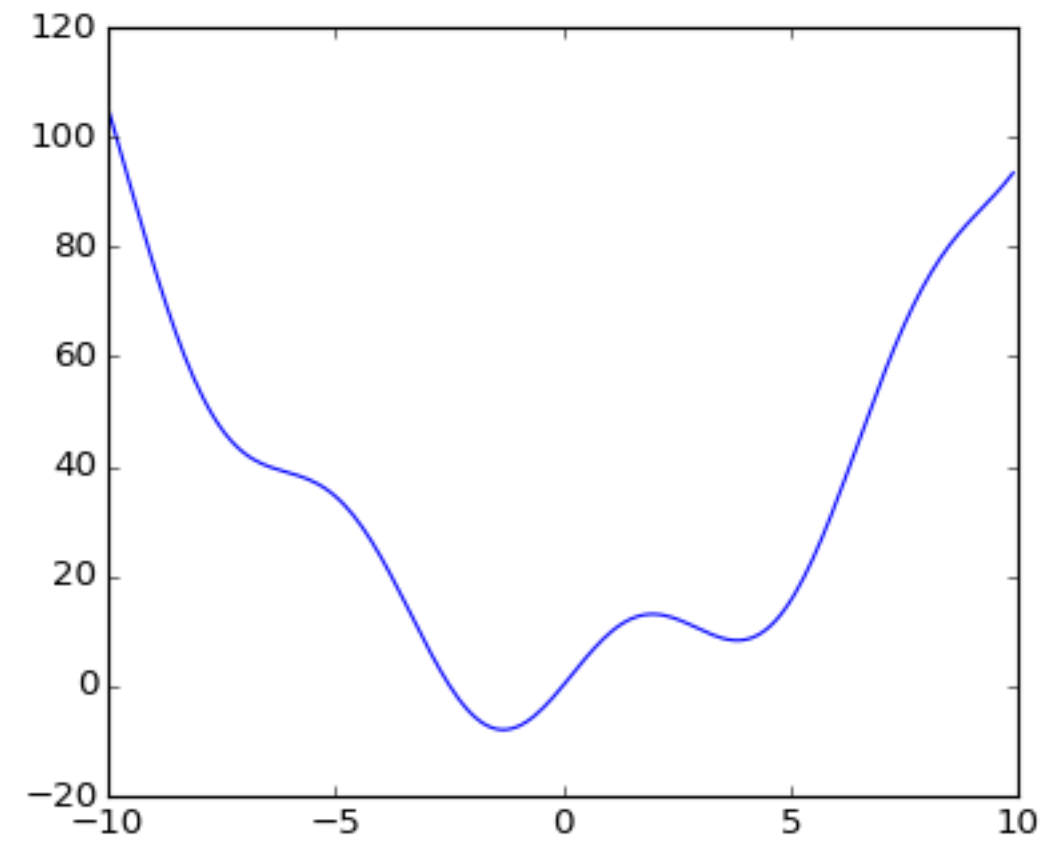


The screenshot shows the SciPy.org website. At the top, there is a blue header with the SciPy logo and the text "SciPy.org" and "Documented by ENTHOUGHT". Below the header, there are navigation links for "Scipy.org" and "Docs". The main content area is titled "SciPy" and includes the following information:

- Release: 1.1.0
- Date: May 05, 2018
- SciPy (pronounced "Sigh Pie") is open-source software for mathematics, science, and engineering.
- Links: Installing and upgrading, API - importing from Scipy, Release Notes
- Tutorial section with a list of submodules: SciPy Tutorial, Introduction, Basic functions, Special functions (scipy.special), Integration (scipy.integrate), Optimization (scipy.optimize), Interpolation (scipy.interpolate), Fourier Transforms (scipy.fftpack), Signal Processing (scipy.signal), Linear Algebra (scipy.linalg), Sparse Eigenvalue Problems with ARPACK, Compressed Sparse Graph Routines (scipy.sparse.csgraph), Spatial data structures and algorithms (scipy.spatial), Statistics (scipy.stats), Multidimensional image processing (scipy.ndimage), File IO (scipy.io)

On the right side, there is a "Table Of Contents" section with links to "SciPy", "Tutorial", "Developer's Guide", and "API Reference". Below that, there is a "Next topic" section with a link to "Installing and upgrading".

SCIPYの例



```
a = np.random.normal(size=1000)
bins = np.arange(-4, 5)
bins
array([-4, -3, -2, -1, 0, 1, 2, 3, 4])

histogram = np.histogram(a, bins=bins, normed=True)[0]
bins = 0.5*(bins[1:] + bins[:-1])
bins
array([-3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5])

from scipy import stats
b = stats.norm.pdf(bins) # norm is a distribution

plt.plot(bins, histogram)

[<matplotlib.lines.Line2D object at ...>]

plt.plot(bins, b)

[<matplotlib.lines.Line2D object at ...>]
```

```
import numpy as np
from scipy import optimize
def f(x):
    return x**2 + 10*np.sin(x)
x = np.arange(-10, 10, 0.1)
plt.plot(x, f(x))
plt.show()
optimize.fmin_bfgs(f, 0)
```

```
Optimization terminated successfully.
Current function value: -7.945823
Iterations: 5
Function evaluations: 24
Gradient evaluations: 8
array([-1.30644003])
```

```
optimize.fmin_bfgs(f, 3, disp=0)
array([ 3.83746663])
optimize.basinhopping(f, 0)
```

```
minimization_failures: 0
nfev: 1725
fun: -7.9458233756152845
x: array([-1.30644001])
message: ['requested number of basinhopping iterations completed successfully']
njev: 575
nit: 100
```

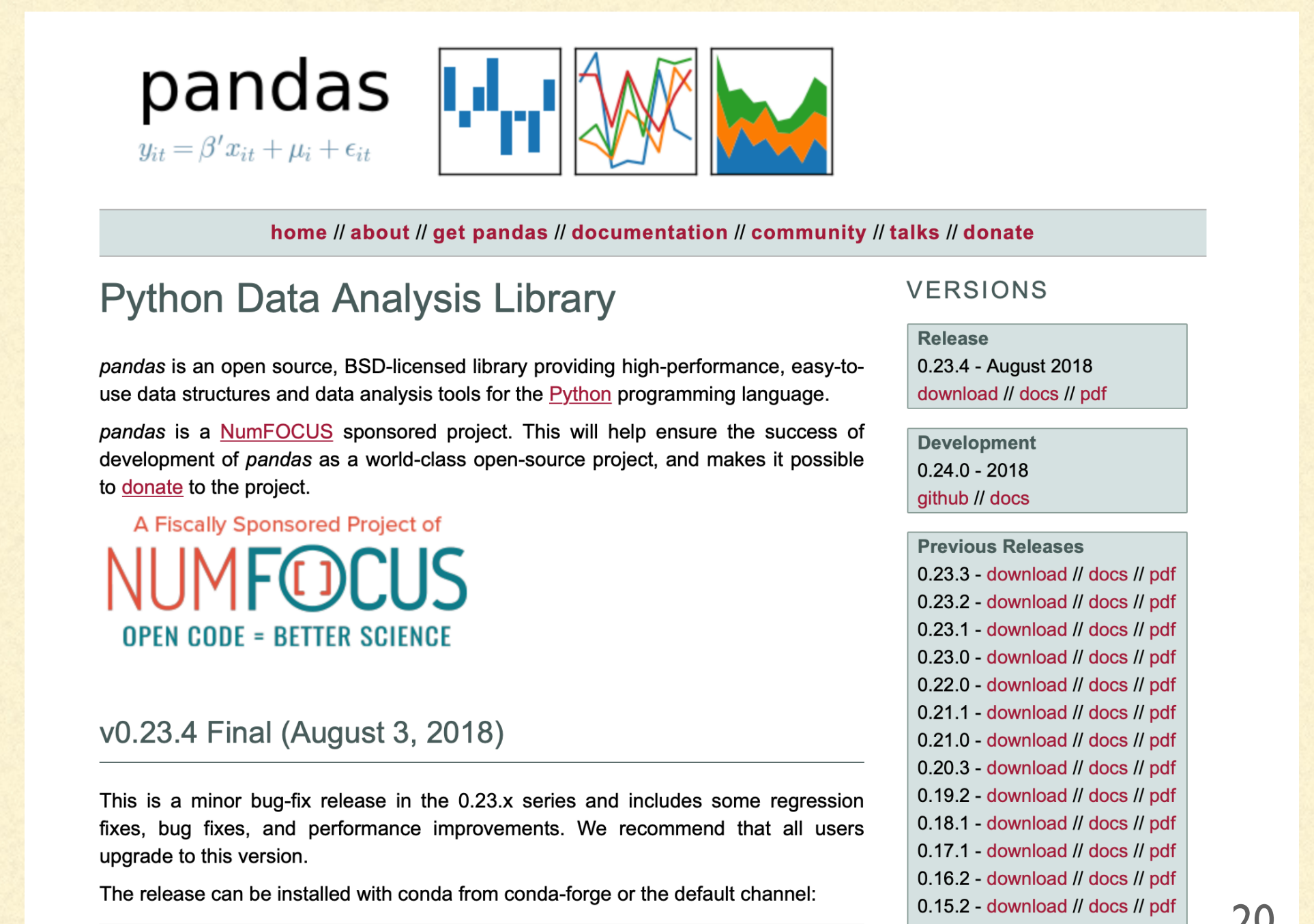
```
xmin_local = optimize.fminbound(f, 0, 10)
xmin_local
3.8374671...
```

PANDAS

- pandasはNumPyをベースに開発されているモジュールです
- Rと似ているデータフレームDataFrameというオブジェクトと、時系列解析のためのシリーズSeriesというオブジェクトを提供します
- Excel形式やCSV形式の入出力が可能です

```
import pandas as pd;
```

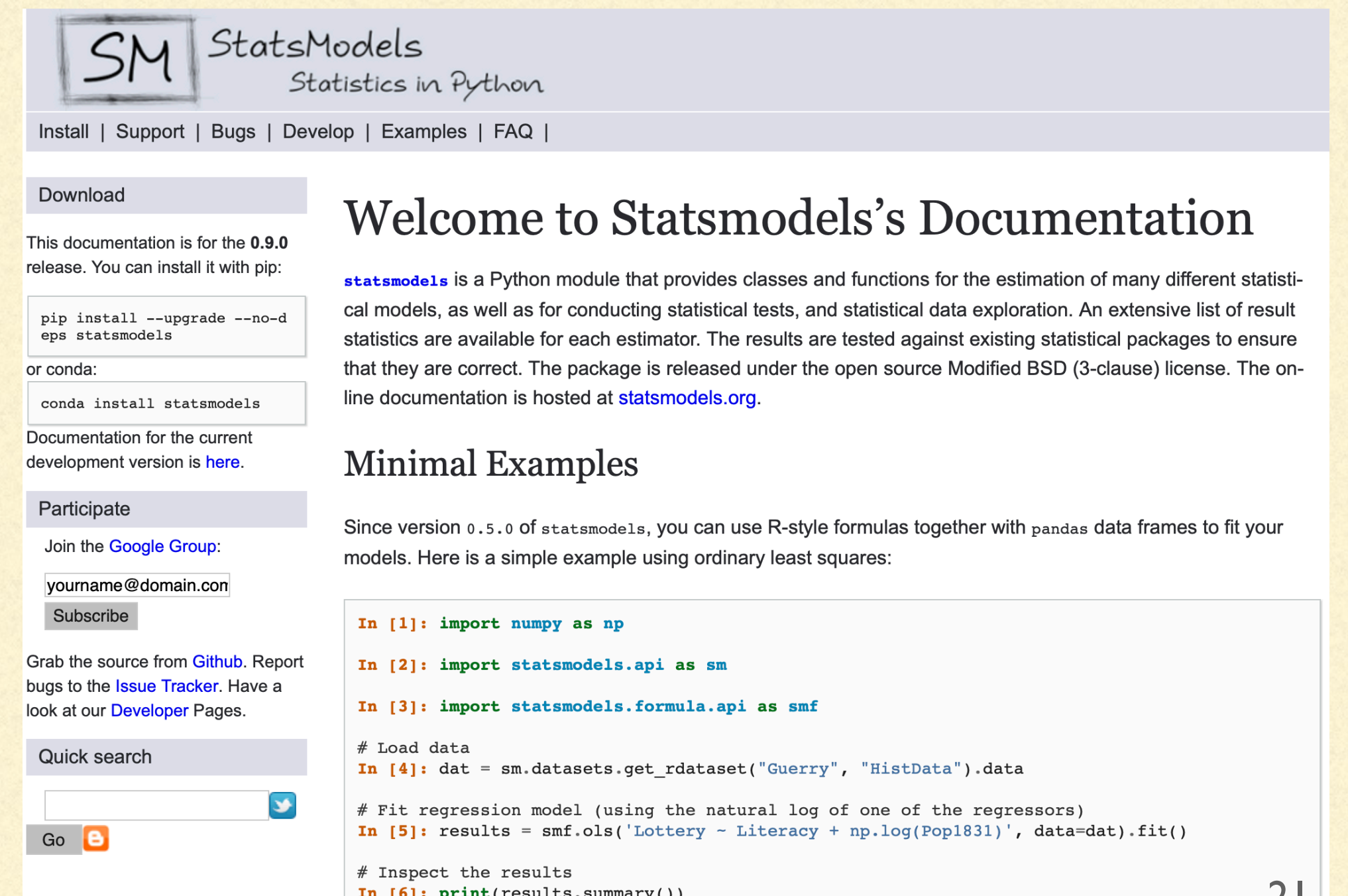
- pandasはpdという名前でもimportされる例を多く見かけます



The screenshot shows the pandas website. At the top, the pandas logo is displayed with the equation $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$ below it. To the right of the logo are three small icons: a bar chart, a line graph, and a stacked area chart. Below the logo is a navigation menu with links: [home](#) // [about](#) // [get pandas](#) // [documentation](#) // [community](#) // [talks](#) // [donate](#). The main heading is "Python Data Analysis Library". Below this, there is a paragraph describing pandas as an open source, BSD-licensed library. Another paragraph mentions that pandas is a NumFOCUS sponsored project. Below the text is the NumFOCUS logo with the tagline "OPEN CODE = BETTER SCIENCE". On the right side, there is a "VERSIONS" section with two boxes: "Release" (0.23.4 - August 2018) and "Development" (0.24.0 - 2018). Below these are "Previous Releases" with a list of versions from 0.23.3 down to 0.14.1, each with links for download, docs, and pdf. At the bottom, there is a section for "v0.23.4 Final (August 3, 2018)" with a paragraph of release notes and a note about installation with conda.

STATSMODELS

- statsmodelsは統計モデルを多く含むモジュールです
- 開発グループのウェブサイトは、トップページに使い方の例題を載せています



SM StatsModels
Statistics in Python

Install | Support | Bugs | Develop | Examples | FAQ |

Download

This documentation is for the **0.9.0** release. You can install it with pip:

```
pip install --upgrade --no-deps statsmodels
```

or conda:

```
conda install statsmodels
```

Documentation for the current development version is [here](#).


Participate

Join the [Google Group](#):

Subscribe

Grab the source from [Github](#). Report bugs to the [Issue Tracker](#). Have a look at our [Developer Pages](#).

Quick search

Go 

Welcome to Statsmodels's Documentation

statsmodels is a Python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration. An extensive list of result statistics are available for each estimator. The results are tested against existing statistical packages to ensure that they are correct. The package is released under the open source Modified BSD (3-clause) license. The on-line documentation is hosted at statsmodels.org.

Minimal Examples

Since version 0.5.0 of **statsmodels**, you can use R-style formulas together with **pandas** data frames to fit your models. Here is a simple example using ordinary least squares:

```
In [1]: import numpy as np
In [2]: import statsmodels.api as sm
In [3]: import statsmodels.formula.api as smf

# Load data
In [4]: dat = sm.datasets.get_rdataset("Guerry", "HistData").data

# Fit regression model (using the natural log of one of the regressors)
In [5]: results = smf.ols('Lottery ~ Literacy + np.log(Pop1831)', data=dat).fit()

# Inspect the results
In [6]: print(results.summary())
```

SCIKIT-LEARN

- scikit-learnは、教師あり学習と教師なし学習のための多くの機械学習の手法を含む、巨大なモジュール群です
 - 分類問題 Classification
 - 回帰問題 Regression
 - クラスタリング Clustering
 - 次元圧縮 Dimension reduction
 - モデル選択 Model selection
 - 前処理 Preprocessing
- これもチュートリアルが充実しています

scikit-learn
Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification
Identifying to which category an object belongs to.
Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression
Predicting a continuous-valued attribute associated with an object.
Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering
Automatic grouping of similar objects into sets.
Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction
Reducing the number of random variables to consider.
Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection
Comparing, validating and choosing parameters and models.
Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing
Feature extraction and normalization.
Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

細かいこと

- 個々のモジュールやメソッドの仕様や使い方については、それぞれのドキュメントを自ら参照することをお勧めします
- どんな言語でもそうですが、同じ名称の関数が、いつのまにか仕様変更となっていることがあるためです